Scalable Two-Phase Flow Solvers

Niall Bootland
Mansfield College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2018

Acknowledgements

Firstly, I would like to sincerely thank my supervisors Prof. Andy Wathen and Dr. Chris Kees, without whom this work would have never come to fruition. Their generous support and guidance have been instrumental throughout the course of this project and their enthusiasm has always encouraged and inspired. Through clear-headed supervision they have helped me to develop the insight and understanding that was invaluable in the pursuit of this thesis.

Secondly, I extend my gratitude to all of the InFoMM team who have done so much to make the doctoral programme a successful and enjoyable experience. Further, to my industrial partners at the US Army Coastal and Hydraulics Laboratory and HR Wallingford for the opportunity to pursue this work and visit new continents.

Thirdly, my thanks go to all the great friends I have made during my time at Oxford who have helped the days pass far too quickly.

Finally, I am indebted to both my parents and grandparents for their immeasurable love and support in all things.

Abstract

Two-phase flows arise in many areas of application such as in the study of coastal and hydraulic processes. Often the fluids involved can be modelled by incompressible phases which have disparate physical properties such as density and viscosity.

We utilise a two-phase flow model of immiscible Newtonian fluids. A key part of this model is a set of variable coefficient Navier–Stokes equations. This thesis focuses on the numerical solution of the linear systems which arise after linearisation of these equations. Solving these systems often dominates the computation time when running simulations.

One of the challenges in solving the model is that the density and viscosity coefficients are discontinuous and can have large jumps between the two phases. In this work we consider preconditioned iterative Krylov methods to solve the large and sparse linear systems and pay particular attention to incorporating the highly varying coefficients into the block preconditioners that we propose. We will see that such considerations can be essential in order to obtain good performance.

An important issue is the scalability of the solution methodology. Here, we will study how the convergence of the iterative solver depends on a grid parameter which controls the refinement of the computational mesh. We will see that the novel preconditioners we propose can lead to convergence which is effectively independent of the grid parameter. We also investigate dependence on other model parameters such as the Reynolds number as well as the density and viscosity ratios between the two fluids.

Another topic we examine is the use of a multipreconditioned iterative method allowing more than one preconditioner to be used simultaneously. Our results using this approach show some promising features.

Finally, we consider an implementation within a more realistic model used in practice for simulating complex air–water flows. In particular, we will provide results for a problem modelling the breaking of a dam.

Contents

Chapter 1

Introduction

1.1 The challenge

In this thesis we shall consider solving equations describing incompressible two-phase flow numerically. We utilise a level set approach to capture the fluid interface between the two phases and finite element methodology for spatial discretisation. The variable coefficient Navier–Stokes equations that model the fluid flow yield a computationally challenging system to solve. It is known that one particular step of the process often dominates the time spent computing: this is solving the linear systems that arise from the Oseen or Newton linearisation. This topic will be the focus of our attention.

For efficient and scalable solution of large linear systems arising in applications, iterative methods must be employed, typically iterative Krylov methods [194]. When modelling incompressible Navier–Stokes flow we arrive at a generalised saddle point problem which, upon discretisation, yields a coefficient matrix of the form

$$K = \begin{pmatrix} F & B^T \\ B & -C \end{pmatrix}. \tag{1.1}$$

Saddle point systems are so called because solutions corresponds to saddle points of a certain Lagrangian function [17]. This requires F to be symmetric, as in the case of the Stokes problem. For the Navier–Stokes equations the advective term contributes an asymmetry in F, as such the system is strictly of a generalised saddle point form (see [48, 17]), also known as a nonsymmetric saddle point form (though sometimes referred to simply as a saddle point form without distinction).

Preconditioning is a crucial requirement for practical and scalable solution of the (generalised) saddle point systems arising in incompressible flow problems. Broadly speaking, a preconditioner can be thought of as a transformation of the linear system to one which is more amenable to iterative solution methods, while still allowing the solution of the original problem to be easily constructed.

For single-phase (constant density and viscosity) Navier–Stokes systems certain block preconditioners based on approximate block factorisations have been shown to be efficient solvers and can exhibit nearly mesh-independent convergence rates for Krylov iterative methods [71, 54]. The primary challenge for development of such block preconditioners is to incorporate a faithful approximation of the Schur complement arising in the factorisation. Moreover, this approximation must allow for an efficient implementation of its use within the block preconditioner. In this work we consider if such preconditioners can be adapted for use when working with models of incompressible two-phase flow.

There has been a large amount of investigation into preconditioning saddle point systems arising in incompressible flow problems. Primarily such work has focused on a single fluid described by the Stokes or Navier–Stokes equations. Approaches to tackle the latter include: domain decomposition [183, 225, 142, 233, 137, 110]; multigrid [238, 252, 116, 250] including algebraic variants [249, 185, 242]; Uzawa-type methods [35]; projection methods [89], null space methods (also called dual variable methods) [176, 98]; constraint preconditioning [82, 28]; incomplete LU factorisation [231, 126], SIMPLE-type methods [232]; and augmented Lagrangian [21, 23] techniques; as well as methods utilising an approximate block factorisation [64, 118, 207, 63, 65, 165, 71]; further, see the reviews [17, 247]. For the Stokes equations there has been further work considering variable coefficient problems [164, 161, 150, 230, 43, 190]. In this case the problem is now symmetric. We will provide a wider and more extensive literature review of solver methodology in Chapter 4.

The Navier–Stokes equations, after appropriate linearisation such as via Picard or Newton iteration, give rise to a nonsymmetric, non-normal system of equations which provides a much more challenging problem in comparison to the Stokes case. To the author's knowledge, very little has been explored for block preconditioning applied to variable viscosity and density Navier–Stokes problems other than recent work in using an augmented Lagrangian approach [100, 101]. To this end, our work pursues methods for the scalable computation of two-phase flow, at the heart of which is the effective preconditioning of two-phase incompressible Navier–Stokes flow.

In particular, we focus on two block preconditioners which utilise an approximate block factorisation. These are the pressure convection–diffusion (PCD) preconditioner [118, 207] and the least-squares commutator (LSC) preconditioner [63, 65]. Both of these methods are comprehensively detailed in [71] for the case of a single fluid with constant density and viscosity. Their consideration for variable coefficient problems is new and, as we shall see, appropriate adaptations are required for two-phase flow.

1.2 Industrial background

Our research is tied closely to that of the US Army Coastal and Hydraulics Laboratory (CHL), and their collaboration with HR Wallingford, both of whom are partnering with the InFoMM CDT. CHL are interested in research into air–water flows which arise in engineering applications such as simulating a dam-break problem, the study of coastal waves, and the designing of channels and coastal structures. While large physical experiments in laboratory tanks are an important part of investigating such flows, they present many challenges when modelling large-scale coastal and hydraulic engineering problems of practical interest. With the advent of increasingly parallel computational facilities to model complex fluid dynamics, state-of-the-art computer simulations are an increasingly important direction of their research.

Instead of modelling only the water and including a moving free surface, and thus a changing domain, the alternative, used in Proteus (software developed by CHL), is to model both the air and water so that a fixed domain can be used and now the air–water interface is tracked. This yields a two-phase Navier–Stokes flow model where the phases are evolved using a level set approach [202, 91, 120]. The interface is marked by the zero level set of a level set function which itself it advected with the flow, allowing the phases to move over time and with the possibility of topological changes.

Problems of interest to CHL are typically high Reynolds number (10^3–10^7) flows computed on very fine meshes (10^6–10^8 vertices). Thus, of particular importance to our research will be to consider how the preconditioners developed depend on the Reynolds number and mesh spacing. We will also explore the effect of different viscosity and density ratios of the two fluids, allowing us to assess robustness to these parameters. Additionally, we will investigate dependence on the size of time-steps taken; for small time-steps the problem becomes dominated by the contribution from the previous time helping to make the problem easier. For very dynamic problems of interest, small time-steps will inevitably be required and in such cases simpler solver techniques can perform reasonably well. However, other problems can result in quasi-steady flow where the free surface may tend toward a steady wake structure or standing wave pattern. Here, larger time-steps can be taken and it is in these more challenging situations that our research may provide the most significant benefits. We will see indications of this when we look towards industrial applications in Proteus in Chapter 9.

1.3 Contribution of the thesis

The primary contribution of this thesis is the exploration of block preconditioning techniques applied to the linear systems arising in computational models of two-phase flow. Our investigation is predominantly computational rather than theoretical and vindicated through numerical results as opposed to rigorous analysis. In part, this is since understanding and quantifying the convergence of iterative Krylov methods applied to highly nonsymmetric and non-normal systems is very challenging, though see [125, 141, 22].

As well as considering existing methodology which can be applied to the linear systems arising in two-phase incompressible Navier–Stokes flow, we develop novel block preconditioners which extend the pressure convection–diffusion (PCD) and the least-squares commutator (LSC) preconditioners to variable coefficient Navier–Stokes systems. While two-phase flow is our main source of test problems, we emphasise that the preconditioning techniques that we will derive apply to systems stemming from more general variable coefficient equations.

Our work also provides several viewpoints on the exposition of the PCD and LSC preconditioners, some of which are not found elsewhere in the literature. These new perspectives along with our careful investigation have enabled us to develop a greater understanding of the underlying approaches and, in particular for the PCD method, allow us to improve performance of the preconditioner even in the single-phase case. Our insight is able to lead us to define appropriate generalisations of these preconditioners for two-phase flow. To assist in this we introduce for the first time the notion of weighted commutators, which are particularly useful in order to understand how a two-phase PCD preconditioner can be constructed.

In our numerical experiments we also consider implementation issues, some of which are scarcely discussed in the literature. For equations which incorporate highly varying coefficients, which may be discontinuous, such issues can become especially important for improving scalability of the overall approach. We provide results to evidence this and confirm our findings. Results will also be given comparing our novel preconditioning techniques with several other methods from the literature. Primarily, our exploration utilises model problems of an academic nature in order to gain a more comprehensive grasp on how the salient features of two-phase flow affect performance. Nonetheless, we begin to study the applicability of our ideas to the more challenging models used in work undertaken at CHL. For this we focus on the PCD approach which is more suited to the discretisation choices made in the Proteus software.

We will observe that preconditioning with our new two-phase PCD approach can yield a scalable solver for the linear systems encountered in our model test problems. In this work, scalability will refer to the number of iterations required by the iterative Krylov method being effectively independent of the refinement of the mesh used, denoted by a mesh scaling parameter h. Scalability will also be seen for an LSC-type approach applied to two-phase flow, however in practice this may be harder to achieve. Further, several challenges still remain for scalable performance within Proteus.

As well as developing new techniques we provide a relatively extensive literature review of preconditioning for incompressible Navier–Stokes flow. Since the body of such literature is vast, we focus on the main approaches used to solve saddle point systems of the form (1.1). A much smaller body of work exists aimed specifically at solving variable coefficient Stokes or Navier–Stokes equations such as those which feature in the modelling of two-phase flow. We highlight the different preconditioners which have been developed in this case, which are predominantly derived and applied for Stokes flow.

We also detail work on utilising a recent multipreconditioning approach which allows us to make use of more than one preconditioner within an iterative Krylov method. Application of the preconditioners can be carried out in parallel and so this approach offers the potential of improved efficiency when computing on parallel architectures, though we do not investigate this here. We provide results applying this multipreconditioning technique to our two-phase flow test problems, considering the combination of two preconditioners. A small modification is introduced to the selective process required to obtain a practical method that ensures the search space does not grow exponentially. This modification allows us to weight the contributions from each preconditioner, allowing us to express a preference if we have knowledge suggesting one of the preconditioners may yield a better choice of search direction. We identify a source of asymmetry in the approach (even with the two preconditioners equally weighted) which has not been discussed before and can result in the ordering of the preconditioners used in the multipreconditioned technique being important. We also explore the impact of having one of the preconditioners being much poorer than the other. We find that inclusion of a much poorer preconditioner can, in some circumstances, still help improve performance and more generally need not cripple the overall performance. This suggests that multipreconditioning can be used to help provide robustness in a simulation with varying flow regimes where different preconditioners may be favourable at different points within the simulations which are unknown in advance.

The outline of the thesis is as follows. We begin with the relevant background work which is developed over three chapters. We first introduce the governing two-phase flow model in Chapter 2. We then discuss discretisation of the model in Chapter 3 and, in particular, consider a mixed finite element formulation. Finally, Chapter 4 outlines relevant iterative solution methods and the requirement of preconditioning before providing an extensive literature review of preconditioning techniques developed for incompressible flow problems.

The main bulk of our work on block preconditioning for two-phase flow is then introduced over two chapters, the primary concern being the derivation of a cheap but faithful approximation to the Schur complement. Each chapter focuses on presenting a particular preconditioner and extending it to the setting of two-phase flow. Chapter 5 develops the pressure convection–diffusion (PCD) approach while Chapter 6 details our work on advancing the least-squares commutator (LSC) methodology. For each approach, selected numerical results are provided to exhibit the effectiveness of the proposed approaches.

A comparison of the two preconditioning techniques we have developed along with other methods from the literature is presented in Chapter 7. We then discuss multipreconditioning in Chapter 8 and provide numerical results for our two-phase test problems. Our focus then turns towards industrial applications in Chapter 9 where we introduce the model used within the Proteus software being developed by CHL, outlining the new challenges it presents, and then apply our two-phase PCD approach to a dam-break test problem and compare it with a simpler method which can perform well for highly dynamic simulations. Finally, conclusions are drawn in Chapter 10 before we outline directions which require further work.

1.4 Acknowledgement of funding

This thesis is based on work supported by the EPSRC Centre for Doctoral Training in Industrially Focused Mathematical Modelling (EP/L015803/1) in collaboration with the US Army Coastal and Hydraulics Laboratory and HR Wallingford.

Chapter 2

Governing model of two-phase flow

The physical process that we model in this chapter is the dynamics of two immiscible fluids which have differing physical properties. In particular, we are interested in flows of two incompressible Newtonian phases, such as water and air[1]. As such, the primary distinction between the fluids will be their density and viscosity.

The modelling of dynamic two-phase flow can be broadly separated into two parts. Firstly, tracking where the two fluids are within the domain over time. Secondly, the fluid dynamics that describes how the phases behave. While these tasks are coupled, when simulating two-phase flow they are typically decoupled at some stage of the computational method so that each is considered independently, even if they are then recombined. Since they present quite different problems we make the distinction here and note that we will ultimately only consider the latter task in detail.

Various methods for tracking multiple phases have been proposed in the literature. The most common are volume-of-fluid (VOF) methods (see e.g. [105, 191, 198, 179]) and level set methods (see e.g. [224, 202, 91, 120, 2]). These two approaches can further be combined (see e.g. [218, 253]). Other techniques also exist, such as front tracking methods [229, 227] and phase field methods [4, 112, 8, 206, 108].

For the majority of this work we will consider a simple level set approach to track the fluids with the fluid dynamics being modelled by a set of variable coefficient incompressible Navier–Stokes equations. In Chapter 9 we consider simulations of real applications where more complicated forms of the governing equations are used, including a mass conserving level set formulation as well as additional stabilisation terms in the Navier–Stokes equations. These details will be discussed in Chapter 9.

[1]Note that, in general, air is compressible. However, at low Mach numbers it is modelled well as an incompressible fluid, as are many other fluids. In our applications of interest the Mach number will be small and so we adopt this simplification here.

Remark 1 (Steady flow) *In the steady case, where the flow is independent of time, the position of the phases may still vary in time. However, even when the positions of the phases remain fixed, and so the task of tracking the phases could be removed, such positions are unlikely to be known in advance for a true steady state when two distinct fluids are present unless results from physical experiments are available. As such, a time-stepping (or pseudo-transient continuation; see Kelley and Keyes [122]) method may still be used. Similar issues arise in free-surface flows. In [234] it is shown that such time integration methods can be inefficient and so a different iterative approach is given which alternates between solving a steady problem with a fixed boundary and updating the free-surface position using a quasi free-surface condition. Convergence of this approach is shown to be asymptotically mesh-independent. We envisage that a similar alternating method could be used for steady two-phase flow, providing an appropriate interface condition is used. Such an approach then requires the solution of a sequence of steady problem with known interface positions. It might also be the case that the two phases are the same fluid but with physical properties (density and viscosity) depending on an external variable and so the interface position may be known, for instance depending on a known temperature profile. More generally this may result in a variable coefficient problem. Our work would also apply in such cases. Note that steady problems are typically more challenging in terms of solving the linear systems that arise while removing complication arising from time-stepping, as such we test our methods on such problems even if the underlying problem may not be physically relevant. Results for fully dynamic simulations reflecting real applications will be given in Chapter 9.*

2.1 A level set method for interface tracking

Let us first establish some notation. Suppose the two-phase problem is defined on an open bounded domain $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$, with one phase occupying Ω_1 and the second Ω_2 such that $\overline{\Omega} = \overline{\Omega}_1 \cup \overline{\Omega}_2$ and $\Omega_1 \cap \Omega_2 = \emptyset$. Let the interface between the two phases be denoted $\Gamma = \partial\Omega_1 \cap \partial\Omega_2$. Note that $\Omega_1(t)$ and $\Omega_2(t)$ may vary over time t, hence so does $\Gamma(t)$, and that neither set is required to be connected. We take the computational domain Ω to be fixed in time and so do not consider any external free-surfaces.

The positions of the two phases are defined through a continuous level set function. For a coordinate vector $\boldsymbol{x}$ we define a scalar function $\phi(\boldsymbol{x}, t)$, called the level set

function, whose zero level set defines the fluid interface $\Gamma(t)$ and so position of the phases. Namely we set

$$\Gamma(t) = \{\boldsymbol{x} \in \Omega \mid \phi(\boldsymbol{x}, t) = 0\}, \tag{2.1a}$$

$$\Omega_1(t) = \{\boldsymbol{x} \in \Omega \mid \phi(\boldsymbol{x}, t) > 0\}, \tag{2.1b}$$

$$\Omega_2(t) = \{\boldsymbol{x} \in \Omega \mid \phi(\boldsymbol{x}, t) < 0\}. \tag{2.1c}$$

Note that such a level set function is not unique but is often given by an approximate signed distance function to the interface Γ. The phases move by being advected by the flow so that, for fluid velocity $\boldsymbol{u}(\boldsymbol{x}, t)$, the evolution of the level set function can be described by

$$\frac{\partial \phi}{\partial t} + \boldsymbol{u} \cdot \nabla \phi = 0. \tag{2.2}$$

The material parameters, namely the density and viscosity, are then prescribed through this level set function. They are given in $\Omega \setminus \Gamma$ by

$$\rho(\phi) = \rho_1 H(\phi) + \rho_2 (1 - H(\phi)), \tag{2.3a}$$

$$\mu(\phi) = \mu_1 H(\phi) + \mu_2 (1 - H(\phi)), \tag{2.3b}$$

where ρ_i and μ_i are, respectively, the density and dynamic viscosity of the fluid which occupy Ω_i, for $i = 1, 2$. The function H is the Heaviside step function, which is 0 for $\phi < 0$ and 1 for $\phi > 0$. Note that, by convention, $H(0) = \frac{1}{2}$. However, this is not required since $\phi = 0$ defines the interface, on which we will provide an interface condition shortly. More generally H could represent a pointwise volume fraction of the first fluid. In numerical simulation, to avoid the discontinuity, a smoothed Heaviside function is often used instead (see e.g. [120]). We emphasise that, in the formation (2.3), the density and viscosity are given by piecewise constants.

2.2 The incompressible Navier–Stokes equations with variable coefficients

The fluid flow, for each of the two phases separately, is modelled by the incompressible Navier–Stokes equations for the velocity $\boldsymbol{u}(\boldsymbol{x}, t)$ and pressure $p(\boldsymbol{x}, t)$ given by

$$\rho_i \frac{\partial \boldsymbol{u}}{\partial t} + \rho_i \, \boldsymbol{u} \cdot \nabla \boldsymbol{u} - \nabla \cdot (2\mu_i \, \boldsymbol{D}\boldsymbol{u}) + \nabla p = \rho_i \boldsymbol{f} \qquad \text{in } \Omega_i, \tag{2.4a}$$

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{in } \Omega_i,\ i = 1, 2. \tag{2.4b}$$

Here we incorporate a body force per unit mass $\boldsymbol{f}$. We use the notation

$$\boldsymbol{D}\boldsymbol{u} = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T\right) \tag{2.5}$$

for the rate-of-deformation tensor (also known as the rate-of-strain tensor) and

$$\boldsymbol{\sigma}(\boldsymbol{u}, p) = -p\,\boldsymbol{I} + 2\mu\,\boldsymbol{D}\boldsymbol{u} \tag{2.6}$$

for the stress tensor, with $\boldsymbol{I}$ being the identity tensor. In these definitions the gradient of a vector field $\boldsymbol{u}$ is a second rank tensor

$$(\nabla \boldsymbol{u})_{i\,j} = \frac{\partial u_i}{\partial x_j}. \tag{2.7}$$

The flow of each phase is coupled by interface (jump) conditions on Γ. We assume continuity of the velocity of the two phases at the interface and a traction boundary condition given respectively by

$$[\boldsymbol{u}]_\Gamma = \boldsymbol{0} \quad \text{on } \Gamma, \tag{2.8a}$$

$$[\boldsymbol{\sigma}\cdot\boldsymbol{n}]_\Gamma = \boldsymbol{g} \quad \text{on } \Gamma, \tag{2.8b}$$

where $\boldsymbol{n}$ is the unit normal to the interface pointing from Ω_1 into Ω_2, $\boldsymbol{\sigma}\cdot\boldsymbol{n}$ is the traction, and $[a]_\Gamma = (a|_{\Omega_1} - a|_{\Omega_2})\,|_\Gamma$ is a jump operator, acting componentwise on vectors. The term $\boldsymbol{g}$ is a localised force term, for instance corresponding to interface tension in the continuum surface force (CSF) model [30, 44, 93], in which case

$$\boldsymbol{g} = \sigma\kappa\boldsymbol{n}, \tag{2.9}$$

where σ is the surface tension coefficient and κ is the mean curvature of the interface Γ. In our case we neglect surface tension and take $\boldsymbol{g} = \boldsymbol{0}$ and so assume a traction-free condition at the interface. Thus (2.8) become the natural jump conditions on Γ, allowing us to consider Navier–Stokes equations in the whole of Ω at once given by

$$\rho(\phi)\,\frac{\partial \boldsymbol{u}}{\partial t} + \rho(\phi)\,\boldsymbol{u}\cdot\nabla\boldsymbol{u} - \nabla\cdot(2\mu(\phi)\,\boldsymbol{D}\boldsymbol{u}) + \nabla p = \rho(\phi)\boldsymbol{f}, \tag{2.10a}$$

$$\nabla\cdot\boldsymbol{u} = 0, \tag{2.10b}$$

with $\rho(\phi)$ and $\mu(\phi)$ defined in (2.3) for some choice of $H(0)$ on the interface. We emphasise that (2.10) are a set of variable coefficient Navier–Stokes equations, in particular with piecewise constant density and viscosity. Since we have discontinuous coefficients it will be necessary to consider the weak formulation of the problem and solutions will be understood in a weak sense; we introduce and discuss the weak

formulation in Chapter 3. When using the CSF model (2.9) the equations can also be combined with the surface tension then appearing as an additional forcing term in the momentum equation (2.10a); see [44] for details.

As well as conditions on the interface, we require boundary conditions on $\partial\Omega$. In this work we consider a 2D domain Ω ($d = 2$); extensions to 3D ($d = 3$) are relatively straightforward. We let $\boldsymbol{n}$ be the outward unit normal to the boundary and $\boldsymbol{t}$ be the unit tangent along the boundary (with an appropriate choice of sign).[2] When considering the weak formulation it is natural to consider four types of condition on the boundary $\partial\Omega$ given by

$$\boldsymbol{u} = \boldsymbol{u}_D \quad \text{on } \partial\Omega_a \quad \text{Dirichlet boundary (inflow),} \tag{2.11a}$$

$$\boldsymbol{\sigma}\cdot\boldsymbol{n} = \boldsymbol{\tau}_D \quad \text{on } \partial\Omega_b \quad \text{traction boundary,} \tag{2.11b}$$

$$\boldsymbol{u}\cdot\boldsymbol{n} = 0 \text{ and } \boldsymbol{t}\cdot\boldsymbol{\sigma}\cdot\boldsymbol{n} = 0 \quad \text{on } \partial\Omega_c \quad \text{free-slip wall,} \tag{2.11c}$$

$$\boldsymbol{u}\cdot\boldsymbol{t} = 0 \text{ and } \boldsymbol{n}\cdot\boldsymbol{\sigma}\cdot\boldsymbol{n} = 0 \quad \text{on } \partial\Omega_d \quad \text{settled channel outflow.} \tag{2.11d}$$

Here, $\boldsymbol{u}_D$ is a known inflow vector along $\partial\Omega_a$ and if $\boldsymbol{u}_D = \boldsymbol{0}$ then we specify a no-slip wall. Further, $\boldsymbol{\tau}_D$ constitutes known external forces on $\partial\Omega_b$ which, in general, gives a free-surface boundary. Since we consider Ω to be fixed we take $\boldsymbol{\tau}_D = \boldsymbol{0}$ giving a traction-free boundary. If we expect the flow to have settled to channel flow on part of the outflow boundary we specify (2.11d), otherwise it can be preferable to specify (2.11b) with $\boldsymbol{\tau}_D = \boldsymbol{0}$ for outflow. We suppose that each section of $\partial\Omega$ is assigned one of the boundary conditions in (2.11).

Note that if $\partial\Omega = \partial\Omega_a \cup \partial\Omega_c$ a compatibility condition

$$\int_{\partial\Omega} \boldsymbol{u}_D\cdot\boldsymbol{n} = 0 \tag{2.12}$$

must be satisfied. This ensures the volume of fluid entering the domain matches the volume of fluid leaving. However, typically (2.11a) is only used for an inflow condition and so $\boldsymbol{u}_D\cdot\boldsymbol{n} \geq 0$, thus one of (2.11b) or (2.11d) must be specified on a non-trivial portion of the boundary $\partial\Omega$. As such, when $\partial\Omega = \partial\Omega_a \cup \partial\Omega_c$ we have that $\boldsymbol{u}_D\cdot\boldsymbol{n} = 0$ on $\partial\Omega_a$ and so consider enclosed flow. In this case the pressure is unique only up to a constant and an additional constraint is needed to fix the level of the pressure. Often this is done by specifying the average pressure, for instance given by enforcing

$$\int_{\Omega} p = 0. \tag{2.13}$$

[2]Note that in 3D there are two tangent vectors to consider.

Finally, in the time-dependent case we suppose that a suitable initial condition $\boldsymbol{u}(\boldsymbol{x}, t_0) = \boldsymbol{u}_0(\boldsymbol{x})$, with $\nabla \cdot \boldsymbol{u}_0 = 0$, is specified. When considering steady flow the time derivative in (2.10a) is set to zero and initial conditions are not necessary, further the level set equation (2.2) is not required, just the position of the interface Γ; see Remark 1.

Remark 2 (Boundary conditions) *Prescribing appropriate and proper boundary conditions within computational fluid dynamics is a non-trivial task and still an open topic of research. The primitive variable formulation of the incompressible Navier–Stokes equations in* (2.4) *is a common choice but alternative formulations have been proposed, partly aimed at avoiding some of the issues arising from boundary conditions and coupling of the relevant physics. A review on boundary conditions and problem formulation can be found in [188]. In the primitive variable formulation we employ we note the importance of keeping the full rate-of-deformation tensor* (2.5) *within* (2.10a) *instead of simplifying the viscous term to a Laplacian formulation; see [136] for a discussion.*

Before moving on we note that the our formulation in (2.4) is a commonly used non-conservative (convection) form of the equations. Alternatively, the conservative form may be considered, in which case convective term $\boldsymbol{u}\cdot\nabla\,\boldsymbol{u}$ is replaced by $\nabla\cdot(\boldsymbol{u}\otimes\boldsymbol{u})$. The types of approach we consider will also be applicable in this case. A different form of the Navier–Stokes equations is the rotation form; see, for instance, [162]. Here, the kinematic pressure p is replaced by the Bernoulli pressure $P = p + \frac{1}{2}\boldsymbol{u}\cdot\boldsymbol{u}$ resulting in governing equations

$$\rho_i \frac{\partial \boldsymbol{u}}{\partial t} + \rho_i \left(\nabla \times \boldsymbol{u}\right) \times \boldsymbol{u} - \nabla \cdot \left(2\mu_i \boldsymbol{D}\boldsymbol{u}\right) + \nabla P = \rho_i \boldsymbol{f} \qquad \text{in } \Omega_i, \tag{2.14a}$$

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{in } \Omega_i,\ i = 1, 2. \tag{2.14b}$$

For this formulation, a slightly different solver methodology should be used [158, 162].

2.3 Nondimensionalisation of the model

We nondimensionalise the governing system of equations describing the fluid flow by considering a characteristic length-scale L and velocity scale U. For this we introduce dimensionless variables using over-bars as follows

$$\begin{aligned} &\boldsymbol{x} = L\bar{\boldsymbol{x}}, \qquad t = \frac{L}{U}\bar{t}, \qquad \boldsymbol{u} = U\bar{\boldsymbol{u}}, \qquad p = \rho_1 U^2 \bar{p}, \qquad \boldsymbol{f} = \frac{U^2}{L}\bar{\boldsymbol{f}}, \\ &\rho = \rho_1\bar{\rho}, \qquad \mu = \mu_1 Re\,\bar{\mu}, \qquad \boldsymbol{u}_D = U\bar{\boldsymbol{u}}_D, \qquad \boldsymbol{u}_0 = U\bar{\boldsymbol{u}}_0, \qquad \phi = L\bar{\phi}, \end{aligned} \tag{2.15}$$

where $Re = LU\rho_1/\mu_1$ is the Reynolds number of the first fluid. Now, immediately dropping the over-bars to avoid notational clutter, the nondimensional form of the governing equations is given by

$$\rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho\, \boldsymbol{u} \cdot \nabla\, \boldsymbol{u} - \nabla \cdot (2\mu\, \boldsymbol{D}\boldsymbol{u}) + \nabla\, p = \rho\, \boldsymbol{f} \qquad \text{in } \Omega, \tag{2.16a}$$

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{in } \Omega, \tag{2.16b}$$

$$\boldsymbol{u} = \boldsymbol{u}_D \qquad \text{on } \partial\Omega_a, \tag{2.16c}$$

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{0} \qquad \text{on } \partial\Omega_b, \tag{2.16d}$$

$$\boldsymbol{u} \cdot \boldsymbol{n} = 0 \text{ and } \boldsymbol{t} \cdot \boldsymbol{\sigma} \cdot \boldsymbol{n} = 0 \qquad \text{on } \partial\Omega_c, \tag{2.16e}$$

$$\boldsymbol{u} \cdot \boldsymbol{t} = 0 \text{ and } \boldsymbol{n} \cdot \boldsymbol{\sigma} \cdot \boldsymbol{n} = 0 \qquad \text{on } \partial\Omega_d, \tag{2.16f}$$

$$\boldsymbol{u} = \boldsymbol{u}_0 \qquad \text{at } t = t_0, \tag{2.16g}$$

along with the level set equation

$$\frac{\partial \phi}{\partial t} + \boldsymbol{u} \cdot \nabla\, \phi = 0 \qquad \text{in } \Omega. \tag{2.17}$$

The level set function defines the regions Ω_1 and Ω_2 and thus the density and viscosity coefficients. After nondimensionalisation ρ and μ in (2.16) take the values

$$\rho = \begin{cases} 1 & \text{in } \Omega_1 \\ \dfrac{\rho_2}{\rho_1} & \text{in } \Omega_2 \end{cases}, \qquad \mu = \begin{cases} \dfrac{1}{Re} & \text{in } \Omega_1 \\ \dfrac{\mu_2}{\mu_1 Re} & \text{in } \Omega_2 \end{cases}, \tag{2.18}$$

so that μ scales inversely with the Reynolds number. Along with suitable boundary and initial conditions for the level set function ϕ, (2.16)–(2.18) form the basis of modelling the incompressible two-phase flow problem.

To proceed in solving (2.16)–(2.18) numerically we will need a discrete form of the system. In particular, we will require a spatial discretisation, for which we will use a mixed finite element method, a way to treat the nonlinearity and coupling within the model, for which we decouple the level set and Navier–Stokes equations and use a Picard or Newton iteration for the nonlinearity in the latter, and a time-stepping procedure to advance in time, for which we consider implicit methods and focus on a simple backwards Euler scheme. This will lead to a generic form of the discrete linearised two-phase Navier–Stokes problem; systems of this form then need to be solved many times in order to simulate flows of interest, hence driving the search for effective preconditioners for these systems.

2.4 Simplifications of the model

In the case of small Reynolds number flows a simplification to the model results in the Stokes equations, as opposed to the Navier–Stokes equations in (2.16a)–(2.16b). For small Reynolds number the viscous term dominates the advective term and a different scaling of the variables, including the pressure, is required compared to that in (2.15). Further, it is common to divide through by the density so that we consider the kinematic viscosity $\nu = \frac{\mu}{\rho}$. The resulting Stokes equations are

$$\frac{\partial \boldsymbol{u}}{\partial t} - \nabla \cdot (2\nu\, \boldsymbol{D}\boldsymbol{u}) + \nabla\, p = \boldsymbol{f}, \tag{2.19a}$$

$$\nabla \cdot \boldsymbol{u} = 0. \tag{2.19b}$$

In the two-phase case the kinematic viscosity is given by

$$\nu = \begin{cases} 1 & \text{in } \Omega_1 \\ \\ \dfrac{\mu_2 \rho_1}{\mu_1 \rho_2} & \text{in } \Omega_2 \end{cases}, \tag{2.20}$$

and, as in the full model above, the level set approach is used to define the two different fluid regions Ω_1 and Ω_2. The Stokes equations (2.19) require similar boundary and initial conditions to the Navier–Stokes equations.

For a single fluid with constant viscosity and density the Stokes equations (2.19) (and in an equivalent manner the Navier–Stokes equations) are most often simplified in the viscous term (in this case with the kinematic viscosity formulation). This uses the impressibility constraint to simplify the divergence of the stress tensor as follows

$$\nabla \cdot (-p\, \boldsymbol{I} + 2\nu\, \boldsymbol{D}\boldsymbol{u}) = -\nabla\, p + \nu\, \nabla \cdot \left(\nabla\, \boldsymbol{u} + (\nabla\, \boldsymbol{u})^T\right) = -\nabla\, p + \nu\, (\nabla \cdot \nabla)\boldsymbol{u} \tag{2.21}$$

since we have

$$\left(\nabla \cdot (\nabla\, \boldsymbol{u})^T\right)_j = \frac{\partial}{\partial x_i}\frac{\partial u_i}{\partial x_j} = \frac{\partial}{\partial x_j}\frac{\partial u_i}{\partial x_i} = \frac{\partial}{\partial x_j}\nabla \cdot \boldsymbol{u} = 0. \tag{2.22}$$

Note that for variable coefficient problems such as two-phase flow we cannot simply pass the divergence through the viscosity ν. The upshot of this simplification is that the viscous term can now be written as a Laplacian and so we obtain, in the Stokes case, the equations

$$\frac{\partial \boldsymbol{u}}{\partial t} - \nu\, (\nabla \cdot \nabla)\, \boldsymbol{u} + \nabla\, p = \boldsymbol{f}, \tag{2.23a}$$

$$\nabla \cdot \boldsymbol{u} = 0. \tag{2.23b}$$

A useful consequence of this Laplacian formulation (2.23) is that now the different components of the velocity are decoupled in the momentum equation (2.23a). This is not true when using the full viscous term, as in (2.19), and the additional coupling for the variable coefficient equations is one of the challenges not typically encountered for single-phase flow. We note, however, that one may need to be cautious in prescribing appropriate boundary conditions for the Laplacian formulation [136].

2.5 Model test problems

In this section we describe some model problems that we use to test our solution techniques. We will use the software package IFISS [209, 70] within MATLAB, which we have adapted to incorporate varying density and viscosity. This allows us to describe dependence on the parameters in a controlled manner. However, there is no capability to update the phases in time. As such we consider solving for a single time-step or else the more challenging case of a steady problem. In each case the fluid interface is prescribed so we need only consider the resulting Navier–Stokes problem. Results for a fully dynamic two-phase problem are presented in Chapter 9 when we discuss moving towards industrial applications. For now we consider two example problems: an enclosed flow problem and an inflow–outflow problem. These model test problems are now described.

Problem 1 (Lid driven cavity) The first model problem we consider is a case of enclosed flow in a square cavity. This is a classic fluid dynamics test problem with the flow being driven by a tangential flow condition on the lid, modelling the flow of an enclosed cavity. We suppose $\Omega = (-1, 1)^2$ and consider a regularised cavity with the prescribed flow along the lid given by

$$\boldsymbol{u} = \begin{pmatrix} 1 - x^4 \\ 0 \end{pmatrix} \quad \text{for} \quad y = 1,\ -1 \leq x \leq 1, \tag{2.24}$$

while no flow boundary conditions ($\boldsymbol{u} = \mathbf{0}$) are imposed on the remaining three edges. As such, all boundary conditions are of Dirichlet type (2.11a). For a two-phase flow problem we consider a "square-in-a-square" configuration with $\Omega_2 = (-\frac{1}{2}, \frac{1}{2})^2$, and thus $\Omega_1 = \Omega \setminus \overline{\Omega}_2$; see Figure 2.1 for a schematic of the set-up. We note that this configuration of fluids is similar to that in the SINKER problem used in [150, 230]. We suppose that no external body forces are acting and neglect surface tension. More details on this test problem in the single-phase case are found in [71].

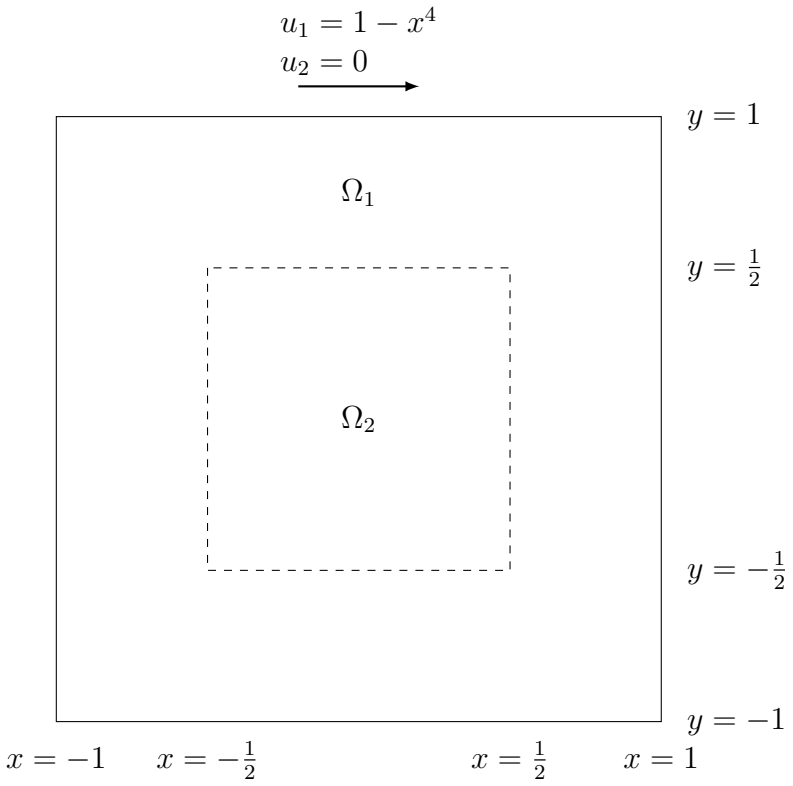


Figure 2.1: Set-up for the enclosed flow lid driven cavity problem (Problem 1). The fluid interface is given by the dotted line and unlabelled boundaries are no-slip walls.

Problem 2 (Backward-facing step) As a second model problem we consider flow over a backward-facing step. A schematic of the set-up is given in Figure 2.2. This problem consists of flow incoming from the left of the domain, flowing over a step, before exiting at the right. The domain is L-shaped, representing a rectangular duct that suddenly expands in height. A parabolic inflow (Poiseuille flow) is imposed on the left, $x = -1$, $0 \leq y \leq 1$, given by a horizontal velocity component $4y(1-y)$ and no vertical velocity. The duct expands to cover $-1 \leq y \leq 1$ at $x = 0$ before a settled channel outflow condition (2.11d) is imposed at $x = 5$, $-1 \leq y \leq 1$. On the top and bottom boundaries the free-slip condition (2.11c) is imposed. Note that, while the Poiseuille inflow is consistent with no-slip channel boundaries, we predominantly use free-slip walls here to incorporate this condition into our test problems and show that the techniques we consider handle such boundaries just as well. We will also consider the default step problem in IFISS which is otherwise identical but replaces the free-slip condition (2.11c) with the no-slip condition $\boldsymbol{u} = \mathbf{0}$ on the top and bottom boundaries and replaces the outflow boundary condition (2.11d) with $\boldsymbol{\sigma} \cdot \boldsymbol{n} = \mathbf{0}$; we label this Problem 2a.

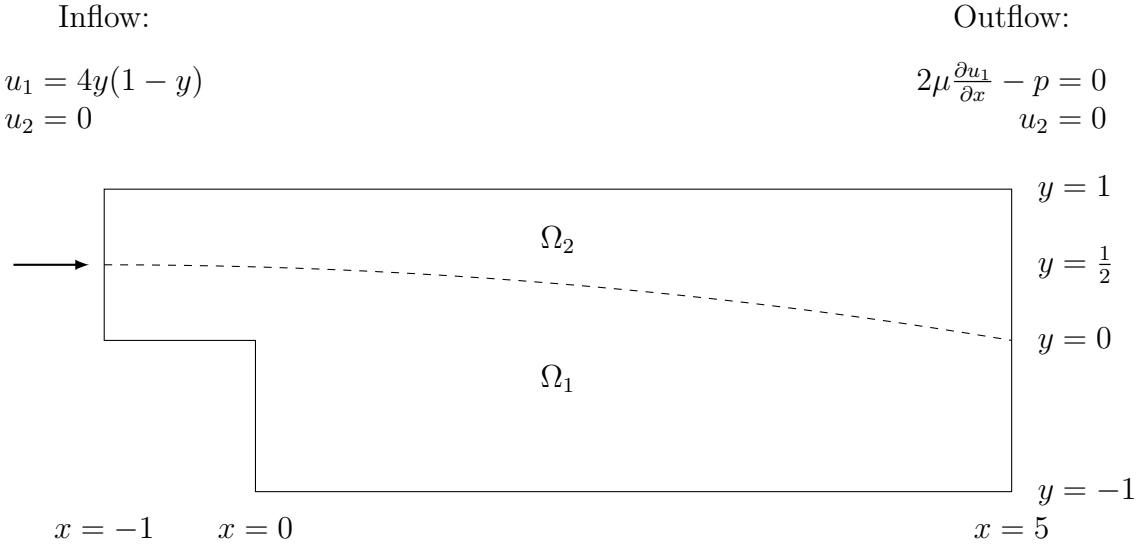


Figure 2.2: Set-up for the inflow–outflow backwards-facing step problem (Problem 2). The fluid interface is given by the dotted line and the unlabelled top and bottom boundaries are free-slip walls.

For our two-phase problem we suppose one fluid (air) is above the other (water) with the boundary being a section of a parabola $y = y_{\text{in}} - \eta_0(x - x_{\text{in}})^2$, where $x_{\text{in}} = -1$ is the x position of the inflow, $y_{\text{in}} = \frac{1}{2}$ is the y position of the phase boundary at the inflow, and η_0 is chosen so that the phase boundary is half-way up the outflow boundary at $x = 5$, $y = 0$. Further, we allow the phase boundary to be a smooth transition of width $\delta = \frac{3}{2}h$ each side of the parabola defining the interface, with h being the grid parameter measuring the side length of the square finite elements that we will use (see Section 3.2). Thus the transition is over a width of three elements, as in [120]. Further, we use the same regularised Heaviside function as [120] to transition, namely

$$H_\delta(s) = \begin{cases} 0 & \text{if } s \le \delta \\ \frac{1}{2}(1 + \frac{s}{\delta} + \frac{1}{\pi}\sin(\frac{s\pi}{\delta})) & \text{if } |s| < \delta \\ 1 & \text{if } s \ge \delta \end{cases}, \tag{2.25}$$

with our argument s being the signed distance function from the parabola. This smoothed form of Heaviside function replaces H in (2.3) to define the variable density and viscosity. For air–water flow we have a density ratio $\widehat{\rho} = \rho_2/\rho_1 = 1.2 \times 10^{-3}$ and a viscosity ratio $\widehat{\mu} = \mu_2/\mu_1 = 1.8 \times 10^{-2}$. We again neglect any surface tension effects and omit any forcing terms.

Remark 3 (Smoothing of the interface) *As well as in the step problem defined in Problem 2, we will consider smoothing of the interface in other model problems including Problem 1 and in the results of Chapter 9. In each case, unless otherwise specified, we allow the transition region the be the width of three elements, namely* $2\delta = 3h$ *where* h *is the relevant measure of element size. Furthermore, we always use the regularised Heaviside function* (2.25) *to transition with* s *being the signed distance function from the interface* Γ *defined by the zero level set of the level set function.*

Chapter 3

Discretisation and mixed finite element formulation

This chapter discusses a numerical solution procedure to solve the governing model equations (2.16)–(2.18). We describe the methodology used to reduce the problem to solving linear systems before focusing on the linearised Navier–Stokes system. Effective preconditioning approaches for solving these linear systems will be the topic of the next chapter.

While many nuances can be encountered in forming appropriate discrete systems (including those addressed in [159, 79, 71, 115, 45, 92, 123]), in this chapter we stick to some of the simplest choices for clarity of our exposition and point out different complications that can arise, some of which will be discussed in later sections.

3.1 Weak formulation

For the mixed finite element framework that we will consider in this work, the natural starting point lies within the weak formulation of the model. Before detailing the weak formulation, we must introduce appropriate function spaces. In what follows, strictly speaking we should use equivalence classes of almost everywhere equal functions. However, for readability we simply refer to 'functions' on this understanding.

3.1.1 Function spaces

The following function spaces will be used within the weak formulation:

Definition 1 *For an open subset $\Omega \subset \mathbb{R}^d$ with piecewise smooth boundary, the space*

of square-integrable functions is defined as

$$L^2(\Omega) = \left\{ u : \Omega \to \mathbb{R} \mid \int_\Omega u^2 < \infty \right\}. \tag{3.1}$$

The associated L^2 inner product is given by

$$(u, v) = \int_\Omega uv \tag{3.2}$$

and induces the L^2 norm $\|u\|_{L^2}^2 = (u, u)$.

Definition 2 *Weak derivatives of a function $u \in L^2(\Omega)$, if they exist, are functions $\partial^\alpha u \in L^2(\Omega)$ for some set of indices $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_d) \in \mathbb{N}_0^d$ that satisfy*

$$\int_\Omega \partial^\alpha u \, \omega = (-1)^{|\alpha|} \int_\Omega u \, \partial^\alpha \omega, \ \forall \, \omega \in C_0^\infty(\Omega), \tag{3.3}$$

where $C_0^\infty(\Omega)$ is the space of infinitely differentiable functions on Ω which are zero on the boundary $\partial\Omega$, $|\alpha| = \sum_{i=1}^d \alpha_i$, and

$$\partial^\alpha \omega = \frac{\partial^{|\alpha|}}{\partial^{\alpha_1} x_1 \partial^{\alpha_2} x_2 \ldots \partial^{\alpha_d} x_d}. \tag{3.4}$$

Definition 3 *A function $u \in L^2(\Omega)$ for which weak derivatives up to and including order n ($|\alpha| \leq n$) exist are said to belong to the Sobolev space $H^n(\Omega)$. In particular we will make use of*

$$H^1(\Omega) = \left\{ u : \Omega \to \mathbb{R} \mid \|u\|_{H^1} < \infty \right\}, \tag{3.5}$$

where the H^1 inner product is given by

$$(u, v)_{H^1} = \int_\Omega uv + \int_\Omega \nabla u \cdot \nabla v \tag{3.6}$$

and induces the H^1 norm $\|u\|_{H^1}^2 = (u, u)_{H^1}$.

Definition 4 *In the weak formulation we will use spaces of vector valued functions. We define $\boldsymbol{H}^1 = [H^1(\Omega)]^2$ as well as the subspace of $\boldsymbol{H}^1$ where the essential boundary conditions of the problem hold*

$$\boldsymbol{H}^1_E = \left\{ \boldsymbol{u} \in \boldsymbol{H}^1 \mid \boldsymbol{u} = \boldsymbol{u}_D \text{ on } \partial\Omega_a, \ \boldsymbol{u} \cdot \boldsymbol{n} = 0 \text{ on } \partial\Omega_c, \ \boldsymbol{u} \cdot \boldsymbol{t} = 0 \text{ on } \partial\Omega_d \right\}, \tag{3.7}$$

which will be the velocity solution space. We also define the vector subspace where homogeneous essential boundary conditions are enforced

$$\boldsymbol{H}^1_{E_0} = \left\{ \boldsymbol{u} \in \boldsymbol{H}^1 \mid \boldsymbol{u} = \boldsymbol{0} \text{ on } \partial\Omega_a, \ \boldsymbol{u} \cdot \boldsymbol{n} = 0 \text{ on } \partial\Omega_c, \ \boldsymbol{u} \cdot \boldsymbol{t} = 0 \text{ on } \partial\Omega_d \right\}, \tag{3.8}$$

which will be the corresponding velocity test function space. We will further make use of $\boldsymbol{L}^2 = [L^2(\Omega)]^2$ and may simply denote the space $L^2(\Omega)$ by L^2 and $H^1(\Omega)$ by H^1 on the understanding that all of the function spaces we consider depend on Ω.

Definition 5 *To incorporate the time-dependent nature we utilise Bochner spaces. The idea is to consider a collection of functions of space that are parametrised by time t. Suppose that we wish to solve over the time interval $[0,T]$. We introduce the space $L^2([0,T],X(\Omega))$ for Sobolev space $X(\Omega)$ depending on space (such as L^2 or H^1) as the space of all Bochner measurable functions $u(t)$ with finite norm*

$$L^2([0,T],X(\Omega)) = \left\{ u : [0,T] \to X(\Omega) \mid \int_0^T \|u(t)\|^2_{X(\Omega)} < \infty \right\}, \tag{3.9}$$

where the function $u(t)$ is Bochner measurable if it equals (almost everywhere) the limit of a sequence of measurably countably-valued functions.

3.1.2 Obtaining the weak formulation

We now consider the weak formulation of model (2.16)–(2.18). To do so we multiply the governing equations by appropriate test functions and integrate. Multiplying the momentum equation (2.16a) by $\boldsymbol{v} \in \boldsymbol{H}^1_{E_0}$ and using integration by parts we have

$$\int_\Omega \rho \left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} \right) \cdot \boldsymbol{v} + \int_\Omega 2\mu \,\mathrm{Tr}\left(\boldsymbol{D}\boldsymbol{u}\,\boldsymbol{D}\boldsymbol{v}\right) - \int_\Omega p \,\nabla \cdot \boldsymbol{v} = \int_\Omega \rho \,\boldsymbol{f} \cdot \boldsymbol{v}. \tag{3.10}$$

Note that the boundary terms from the integration by parts, namely

$$-\int_{\partial\Omega} 2\mu \,\boldsymbol{v} \cdot \boldsymbol{D}\boldsymbol{u} \cdot \boldsymbol{n} + \int_{\partial\Omega} p \,\boldsymbol{v} \cdot \boldsymbol{n} = -\int_{\partial\Omega} \boldsymbol{v} \cdot \boldsymbol{\sigma} \cdot \boldsymbol{n}, \tag{3.11}$$

vanish either due to the boundary conditions on the traction $\boldsymbol{\sigma}\cdot\boldsymbol{n}$ or the homogeneous essential boundary conditions on $\boldsymbol{v}$ by our choice of test function space (3.8). Further the term $\mathrm{Tr}\left(\boldsymbol{D}\boldsymbol{u}\,\boldsymbol{D}\boldsymbol{v}\right)$ is equivalent to $\boldsymbol{D}\boldsymbol{u} : \boldsymbol{D}\boldsymbol{v}$ and is given in index notation as

$$\mathrm{Tr}\left(\boldsymbol{D}\boldsymbol{u}\,\boldsymbol{D}\boldsymbol{v}\right) = \frac{\partial u_i}{\partial x_j}\frac{\partial v_j}{\partial x_i}, \tag{3.12}$$

where we have used the Einstein summation convention. For the incompressibility constraint (2.16b) we multiply by a test function $q \in L^2$ and integrate obtaining

$$\int_\Omega q \,\nabla \cdot \boldsymbol{u} = 0. \tag{3.13}$$

For the level set equation (2.17) we take a test function $\vartheta \in H^1$ (we consider H^1 rather than L^2 to ensure we can apply a streamline diffusion method later) to yield

$$\int_\Omega \left(\frac{\partial \phi}{\partial t} + \boldsymbol{u} \cdot \nabla \,\phi \right) \vartheta = 0. \tag{3.14}$$

It is convenient to write the weak formulation in terms of bilinear and trilinear forms. To this end we introduce the following forms

$$m : \boldsymbol{L}^2 \times \boldsymbol{L}^2 \to \mathbb{R}, \qquad m(\boldsymbol{u}, \boldsymbol{v}) = \int_\Omega \rho(\phi)\, \boldsymbol{u} \cdot \boldsymbol{v}, \tag{3.15}$$

$$n : \boldsymbol{H}^1 \times \boldsymbol{H}^1 \times \boldsymbol{H}^1 \to \mathbb{R}, \qquad n(\boldsymbol{z}; \boldsymbol{u}, \boldsymbol{v}) = \int_\Omega \rho(\phi)\, (\boldsymbol{z} \cdot \nabla\, \boldsymbol{u}) \cdot \boldsymbol{v}, \tag{3.16}$$

$$a : \boldsymbol{H}^1 \times \boldsymbol{H}^1 \to \mathbb{R}, \qquad a(\boldsymbol{u}, \boldsymbol{v}) = \int_\Omega 2\mu(\phi)\, \mathrm{Tr}\left(\boldsymbol{D}\boldsymbol{u}\, \boldsymbol{D}\boldsymbol{v}\right), \tag{3.17}$$

$$b : L^2 \times \boldsymbol{H}^1 \to \mathbb{R}, \qquad b(p, \boldsymbol{u}) = -\int_\Omega p\, \nabla \cdot \boldsymbol{u}. \tag{3.18}$$

Note that the bilinear forms $m(\cdot\,,\cdot)$ and $a(\cdot\,,\cdot)$ and the trilinear form $n(\cdot\,;\cdot\,,\cdot)$ depend on the level set function ϕ.

Using the above definitions, the weak formulation is to find $\boldsymbol{u} \in L^2([0,T], \boldsymbol{H}^1_E)$, $p \in L^2([0,T], L^2)$ and $\phi \in L^2([0,T], H^1)$ such that

$$m\left(\frac{\partial \boldsymbol{u}}{\partial t}, \boldsymbol{v}\right) + n(\boldsymbol{u}; \boldsymbol{u}, \boldsymbol{v}) + a(\boldsymbol{u}, \boldsymbol{v}) + b(p, \boldsymbol{v}) = m(\boldsymbol{f}, \boldsymbol{v}), \tag{3.19a}$$

$$b(q, \boldsymbol{u}) = 0, \tag{3.19b}$$

$$\left(\frac{\partial \phi}{\partial t}, \vartheta\right) + (\boldsymbol{u} \cdot \nabla\, \phi, \vartheta) = 0 \tag{3.19c}$$

for all $\boldsymbol{v} \in \boldsymbol{H}^1_{E_0}$, $q \in L^2$ and $\vartheta \in H^1$. Note the change of sign from (3.13) to (3.19b) which will later provide symmetry of the off-diagonal blocks. We note that the weak formulation of the model problem can be found in the literature, for example in [91].

3.2 Mixed finite elements

To discretise in space we consider a mixed finite element method. In particular, using standard Galerkin discretisations. The main idea is to restrict to finite dimensional subspaces based on a mesh of elements. We can then use bases of these spaces to reduce to an algebraic problem. We suppose that we have a polygonal domain Ω which is split up into a number of subdomains (called elements) on which we define local function spaces which are then pieced together to create a global function space. We consider elements having the same shape (triangles or quadrilaterals) and local function spaces defined by low order polynomials. We denote the (global) finite dimensional subspaces (finite element spaces) by $\mathbb{V}^h \subset \boldsymbol{H}^1_E$, $\mathbb{Q}^h \subset L^2$ and $\mathbb{W}^h \subset H^1$ for the velocity, pressure and level set function respectively. These finite element spaces cannot be arbitrary and must be appropriate for the problem at hand, we will discuss this later in Sections 3.6 and 3.7.

Equipped with these finite element spaces, the discrete variational problem is then to find $\boldsymbol{u}_h \in L^2([0,T], \mathbb{V}^h)$, $p_h \in L^2([0,T], \mathbb{Q}^h)$ and $\phi_h \in L^2([0,T], \mathbb{W}^h)$ such that

$$m\left(\frac{\partial \boldsymbol{u_h}}{\partial t}, \boldsymbol{v}_h\right) + n(\boldsymbol{u}_h; \boldsymbol{u}_h, \boldsymbol{v}_h) + a(\boldsymbol{u}_h, \boldsymbol{v}_h) + b(p_h, \boldsymbol{v}_h) = m(\boldsymbol{f}, \boldsymbol{v}_h), \tag{3.20a}$$

$$b(q_h, \boldsymbol{u}_h) = c(p_h, q_h), \tag{3.20b}$$

$$\left(\frac{\partial \phi_h}{\partial t}, \vartheta_h\right) + (\boldsymbol{u}_h \cdot \nabla\, \phi_h, \vartheta_h) = 0 \tag{3.20c}$$

for all $\boldsymbol{v}_h \in \mathbb{V}_0^h$, $q_h \in \mathbb{Q}^h$ and $\vartheta_h \in \mathbb{W}^h$, where $\mathbb{V}_0^h \subset \boldsymbol{H}^1_{E_0}$ is the same finite element subspace as $\mathbb{V}^h$ except with homogeneous essential boundary conditions enforced. The form $c(\cdot,\cdot)$ may be necessary for stability depending on the choice of finite element spaces used and whether they satisfy the so-called *inf–sup* condition; see Section 3.6

Note that the discrete formulation in (3.20c) for the level set equation turns out to be unstable. However, a streamline diffusion technique can be used to provide stability, we follow the description provided in [91]; this gives a Petrov–Galerkin method. Letting $\mathcal{T}_h$ be the set of elements used, then for each element $T \in \mathcal{T}_h$ a choice of stabilisation parameter δ_T is made. Then elementwise, on each element T, the test functions are chosen as

$$\widehat{\vartheta}_h = \vartheta_h + \delta_T\, \boldsymbol{u}_h \cdot \nabla\, \vartheta_h, \tag{3.21}$$

for $\vartheta_h \in \mathbb{W}^h$. Thus (3.20c) is replaced by

$$\sum_{T\in\mathcal{T}_h} \left(\frac{\partial \phi_h}{\partial t}, \vartheta_h + \delta_T\, \boldsymbol{u}_h \cdot \nabla\, \vartheta_h\right)_T + \sum_{T\in\mathcal{T}_h} (\boldsymbol{u}_h \cdot \nabla\, \phi_h, \vartheta_h + \delta_T\, \boldsymbol{u}_h \cdot \nabla\, \vartheta_h)_T = 0, \tag{3.22}$$

where $(\cdot,\cdot)_T$ is the standard L^2 inner product over T. For further details, such as the choice of stabilisation parameter, see [189]. Note also that for advection dominated flow regimes (high Reynolds number) the advection term $n(\cdot\,;\cdot,\cdot)$ in (3.20a) may also require stabilisation, which might also be done using a streamline diffusion technique; see [207, 159, 79]. In Proteus, a different stabilisation technique is used, this will be discussed further in Chapter 9.

In general, the larger the Reynolds number (inversely proportional to μ) the more challenging the flow problem is to solve and issues such as advective stabilisation and turbulence modelling become important. As such, it is implicit that the discrete model we consider is only appropriate within a certain range of Reynolds numbers. Another potential issue is that the equations are nonlinear and so there need not be a unique solution (we will discuss linearisation shortly), in particular for the steady

case a unique solution may not exist if the Reynolds number exceeds a critical value [129, 221, 222, 80]. We will not worry about these complications in this work.

To identify the corresponding algebraic problem we derive a representation using basis functions for the finite element spaces. Let $\{\boldsymbol{\varphi}_j\}$ be a set of velocity nodal basis functions such that

$$\boldsymbol{u}_h = \sum_{j=1}^{n_u} \mathrm{u}_j \, \boldsymbol{\varphi}_j + \sum_{j=n_u+1}^{n_u+n_\partial} \mathrm{u}_j \, \boldsymbol{\varphi}_j, \tag{3.23}$$

with $\sum_{j=1}^{n_u} \mathrm{u}_j \, \boldsymbol{\varphi}_j \in \mathbb{V}_0^h$. In the second term of (3.23), for the n_∂ nodal basis functions centred on the boundary $\partial\Omega_D$, we fix the coefficients u_j for $j = n_u + 1, \ldots, n_u + n_\partial$ to interpolate the boundary data $\boldsymbol{u}_D$. Similarly, let $\{\psi_j\}$ be a set of pressure nodal basis functions such that

$$p_h = \sum_{j=1}^{n_p} \mathrm{p}_j \, \psi_j, \tag{3.24}$$

and $\{\chi_j\}$ be a set nodal basis functions for the level set function such that

$$\phi_h = \sum_{j=1}^{n_\phi} \Phi_j \, \chi_j. \tag{3.25}$$

Since $\{\boldsymbol{\varphi}_i\}_{1 \le i \le n_u}$, $\{\psi_i\}_{1 \le i \le n_p}$ and $\{\chi_i\}_{1 \le i \le n_\phi}$ are bases for the space of test functions $\boldsymbol{v}_h$, q_h and ϑ_h it is sufficient to consider the cases $\boldsymbol{v}_h = \boldsymbol{\varphi}_i$, $q_h = \psi_i$ and $\vartheta_h = \chi_i$. As such, we construct the following matrices

$$M_{i,j}^{(\rho)}(\phi_h) = \int_\Omega \rho(\phi_h) \, \boldsymbol{\varphi}_j \cdot \boldsymbol{\varphi}_i, \tag{3.26a}$$

$$N_{i,j}^{(\rho)}(\boldsymbol{u}_h, \phi_h) = \int_\Omega \rho(\phi_h) \left(\boldsymbol{u}_h \cdot \nabla \, \boldsymbol{\varphi}_j \right) \cdot \boldsymbol{\varphi}_i, \tag{3.26b}$$

$$A_{i,j}^{(\mu)}(\phi_h) = \int_\Omega 2\mu(\phi_h) \, \mathrm{Tr} \left(\boldsymbol{D}\boldsymbol{\varphi}_j \, \boldsymbol{D}\boldsymbol{\varphi}_i \right), \tag{3.26c}$$

$$B_{i,j} = -\int_\Omega \psi_i \, \nabla \cdot \boldsymbol{\varphi}_j, \tag{3.26d}$$

$$C_{i,j} = c(\psi_j, \psi_i), \tag{3.26e}$$

$$E_{i,j}(\boldsymbol{u}_h) = \sum_{T \in \mathcal{T}_h} \int_T \chi_j \left(\chi_i + \delta_T \, \boldsymbol{u}_h \cdot \nabla \, \chi_i \right), \tag{3.26f}$$

$$H_{i,j}(\boldsymbol{u}_h) = \sum_{T \in \mathcal{T}_h} \int_T \left(\boldsymbol{u}_h \cdot \nabla \, \chi_j \right) \left(\chi_i + \delta_T \, \boldsymbol{u}_h \cdot \nabla \, \chi_i \right), \tag{3.26g}$$

and vectors

$$\begin{aligned}\mathrm{f}_i(\boldsymbol{u}_h,\phi_h) = &\int_\Omega \rho(\phi_h)\,\boldsymbol{f}\cdot\boldsymbol{\varphi}_i \;-\sum_{j=n_u+1}^{n_u+n_\partial}\frac{\partial \mathrm{u}_j}{\partial t}\int_\Omega \rho(\phi_h)\,\boldsymbol{\varphi}_j\cdot\boldsymbol{\varphi}_i\\ &-\sum_{j=n_u+1}^{n_u+n_\partial}\mathrm{u}_j\int_\Omega 2\mu(\phi_h)\,\mathrm{Tr}\left(\boldsymbol{D}\boldsymbol{\varphi}_j\,\boldsymbol{D}\boldsymbol{\varphi}_i\right) \qquad (3.27a)\\ &-\sum_{j=n_u+1}^{n_u+n_\partial}\mathrm{u}_j\int_\Omega \rho(\phi_h)\left(\boldsymbol{u}_h\cdot\nabla\,\boldsymbol{\varphi}_j\right)\cdot\boldsymbol{\varphi}_i,\\ \mathrm{g}_i(\boldsymbol{u}_h) = &\sum_{j=n_u+1}^{n_u+n_\partial}\mathrm{u}_j\int_\Omega \psi_i\,\nabla\cdot\boldsymbol{\varphi}_j. \qquad (3.27b)\end{aligned}$$

We have used the notation of bracketed exponents when we wish to explicitly illustrate the scaling by density and viscosity within the matrices in (3.26). The stabilisation matrix C, which should be positive semi-definite if not zero, will depend on the choice of finite elements used and, as such, we defer precise details until discussing such choices in Section 3.7; note that C might also incorporate some scaling, typically with the inverse of viscosity.

We collect together the unknowns for velocity, pressure and level set function values as

$$\mathbf{u} = (\mathrm{u}_1,\ldots,\mathrm{u}_{n_u})^T, \qquad (3.28a)$$

$$\mathbf{p} = (\mathrm{p}_1,\ldots,\mathrm{p}_{n_p})^T, \qquad (3.28b)$$

$$\boldsymbol{\Phi} = (\Phi_1,\ldots,\Phi_{n_\phi})^T, \qquad (3.28c)$$

and knowns as

$$\mathbf{f} = (\mathrm{f}_1,\ldots,\mathrm{f}_{n_u})^T, \qquad (3.29a)$$

$$\mathbf{g} = (\mathrm{g}_1,\ldots,\mathrm{g}_{n_p})^T, \qquad (3.29b)$$

$$\mathbf{u}_D = (\mathrm{u}_{n_u+1},\ldots,\mathrm{u}_{n_u+n_\partial})^T. \qquad (3.29c)$$

For consistency with the number of arguments given in the definitions (3.26)–(3.27), we also let $\mathbf{u}_{bc} = (\mathbf{u}^T, \mathbf{u}_D^T)^T$ collect together all velocity values. This allows us to write the discrete variational problem as: find $\mathbf{u}\in\mathbb{R}^{n_u}$, $\mathbf{p}\in\mathbb{R}^{n_p}$ and $\boldsymbol{\Phi}\in\mathbb{R}^{n_\phi}$ such that

$$M^{(\rho)}(\boldsymbol{\Phi})\frac{\partial \mathbf{u}}{\partial t} + N^{(\rho)}(\mathbf{u}_{bc},\boldsymbol{\Phi})\mathbf{u} + A^{(\mu)}(\boldsymbol{\Phi})\mathbf{u} + B^T\mathbf{p} = \mathbf{f}(\mathbf{u}_{bc},\boldsymbol{\Phi}), \qquad (3.30a)$$

$$B\mathbf{u} - C\mathbf{p} = \mathbf{g}(\mathbf{u}_{bc}), \qquad (3.30b)$$

$$E(\mathbf{u}_{bc})\frac{\partial \boldsymbol{\Phi}}{\partial t} + H(\mathbf{u}_{bc})\boldsymbol{\Phi} = 0. \qquad (3.30c)$$

Note that we include the discrete variables that the matrices and right-hand side vectors depend on in brackets, the continuous variables are given through (3.23) and (3.25). System (3.30) forms a set of nonlinear coupled ordinary differential equations to solve.

3.3 Time-stepping

We now consider discretisation in time, for this we detail a simple Implicit–Explicit (IMEX) scheme for time-stepping. Let the discrete variables at time $t^m = t^{m-1} + \Delta t$ be denoted by $\mathbf{u}^m$, $\mathbf{p}^m$ and $\mathbf{\Phi}^m$. Note that (3.30) contains couplings between the level set and Navier–Stokes equations, by lagging the level set function in the coefficients matrices of the Navier–Stokes problem we can remove this nonlinear coupling and so treat these coefficients explicitly. For the remaining terms, a backwards Euler scheme yields approximate solutions implicitly as

$$\begin{aligned} M^{(\rho)}(\mathbf{\Phi}^{m-1})\frac{\mathbf{u}^m - \mathbf{u}^{m-1}}{\Delta t} + N^{(\rho)}((\mathbf{u}_{bc})^m, \mathbf{\Phi}^{m-1})\mathbf{u}^m \\ + A^{(\mu)}(\mathbf{\Phi}^{m-1})\mathbf{u}^m + B^T\mathbf{p}^m = \mathbf{f}((\mathbf{u}_{bc})^m, \mathbf{\Phi}^{m-1}), \end{aligned} \tag{3.31a}$$

$$B\mathbf{u}^m - C\mathbf{p}^m = \mathbf{g}((\mathbf{u}_{bc})^m), \tag{3.31b}$$

$$E((\mathbf{u}_{bc})^m)\frac{\mathbf{\Phi}^m - \mathbf{\Phi}^{m-1}}{\Delta t} + H((\mathbf{u}_{bc})^m)\mathbf{\Phi}^m = 0. \tag{3.31c}$$

Note that in this time-stepping scheme we first solve the Navier–Stokes portion of the system (3.31a)–(3.31b) to compute $\mathbf{u}^m$, $\mathbf{p}^m$ based on the previous level set function $\mathbf{\Phi}^{m-1}$, we then update the level set function to give $\mathbf{\Phi}^m$ based on the newly computed $\mathbf{u}^m$ in (3.31c). This time-stepping scheme can be found in, for instance, [120].

More advanced methods can be used for time-stepping, for example the fractional θ-schemes described in [40, 91]. Further exposition of other time-stepping methods and adaptive time-step control for the single-phase problems are given in [71]. Though improved time-stepping is possible and a current area of research, we stick here to a simple scheme to demonstrate the approach. To reduce (3.31) solely to solving linear systems, we finally require a nonlinear iteration for the advective term. Note that the discrete level set equation (3.31c) is now decoupled[1] and linear, the majority of the work is in solving the (coupled) discrete Navier–Stokes equations (3.31a)–(3.31b), as such we focus on this set of equations.

[1]There is still a one-way coupling present so (3.31c) cannot be solved completely independently.

3.4 Nonlinear iteration

Two standard approaches for treating the nonlinearity in the Navier–Stokes equations are Picard linearisation and Newton linearisation; we detail both approaches. In each case we start with initial iterates (which may be the solution from the previous time-step or from solving the corresponding Stokes problem) and use an iterative procedure to update until an appropriate convergence criterion is satisfied.

3.4.1 Picard linearisation

When using Picard iteration we lag the velocity component $\mathbf{u}^m$ required in the advection matrix $N^{(\rho)}$ (and the associated boundary term in $\mathbf{f}$) in (3.31a). If we use an index k for the nonlinear iteration and, in solving (3.31a)–(3.31b), start with initial iterates $\mathbf{u}_0^m$ and $\mathbf{p}_0^m$ then Picard iteration solves

$$M^{(\rho)}(\mathbf{\Phi}^{m-1})\frac{\mathbf{u}_k^m - \mathbf{u}^{m-1}}{\Delta t} + N^{(\rho)}((\mathbf{u}_{bc})_{k-1}^m, \mathbf{\Phi}^{m-1})\mathbf{u}_k^m + A^{(\mu)}(\mathbf{\Phi}^{m-1})\mathbf{u}_k^m + B^T\mathbf{p}_k^m = \mathbf{f}((\mathbf{u}_{bc})_{k-1}^m, \mathbf{\Phi}^{m-1}), \tag{3.32a}$$

$$B\mathbf{u}_k^m - C\mathbf{p}_k^m = \mathbf{g}((\mathbf{u}_{bc})^m), \tag{3.32b}$$

for $k = 1, 2, \ldots$ until suitable convergence has been achieved. Note that $\mathbf{u}^{m-1}$ is from the previous time-step and so is already known and that the initial pressure iterate $\mathbf{p}_0^m$ is not actually required. Picard linearisation of the Navier–Stokes equations results in an Oseen system (3.32) where the wind is given by the previous Picard iterate defined by $\mathbf{u}_k^m$.

The Oseen system corresponds to a fixed-point iteration and it can be shown the iteration contracts and thus the method is globally convergent, independent of the initial iterate, when a unique solution exists; see [117]. In general Picard iteration has linear convergence whereas Newton iteration has quadratic convergence. We now discuss this latter process.

3.4.2 Newton linearisation

Newton iteration can be thought of as using a Taylor expansion based on the current iterate, dropping quadratic and higher order terms to solve the remaining linear model. Typically this is considered as providing Newton correction terms $\delta\mathbf{u}_k^m$ and $\delta\mathbf{p}_k^m$ to the previous iterates to yield $\mathbf{u}_k^m = \mathbf{u}_{k-1}^m + \delta\mathbf{u}_k^m$ and $\mathbf{p}_k^m = \mathbf{p}_{k-1}^m + \delta\mathbf{p}_k^m$. The

Newton corrections satisfy

$$\begin{aligned} \Delta t^{-1} M^{(\rho)}(\mathbf{\Phi}^{m-1})\delta\mathbf{u}_k^m + N^{(\rho)}((\mathbf{u}_{bc})_{k-1}^m, \mathbf{\Phi}^{m-1})\delta\mathbf{u}_k^m \\ + W^{(\rho)}((\mathbf{u}_{bc})_{k-1}^m, \mathbf{\Phi}^{m-1})\delta\mathbf{u}_k^m + A^{(\mu)}(\mathbf{\Phi}^{m-1})\delta\mathbf{u}_k^m + B^T\delta\mathbf{p}_k^m = \mathbf{r}_{k-1}, \end{aligned} \tag{3.33a}$$

$$B\delta\mathbf{u}_k^m - C\delta\mathbf{p}_k^m = \mathbf{s}_{k-1}, \tag{3.33b}$$

where (in the notation of (3.26))

$$W_{i,j}^{(\rho)}(\boldsymbol{u}_h, \phi_h) = \int_\Omega \rho(\phi_h)\left(\boldsymbol{\varphi}_j \cdot \nabla\, \boldsymbol{u}_h\right) \cdot \boldsymbol{\varphi}_i, \tag{3.34}$$

and the nonlinear residuals are given by

$$\begin{aligned} \mathbf{r}_{k-1} = \mathbf{f}((\mathbf{u}_{bc})_{k-1}^m, \mathbf{\Phi}^{m-1}) + \Delta t^{-1} M^{(\rho)}(\mathbf{\Phi}^{m-1})\mathbf{u}^{m-1} - \Delta t^{-1} M^{(\rho)}(\mathbf{\Phi}^{m-1})\mathbf{u}_{k-1}^m \\ - N^{(\rho)}((\mathbf{u}_{bc})_{k-1}^m, \mathbf{\Phi}^{m-1})\mathbf{u}_{k-1}^m - A^{(\mu)}(\mathbf{\Phi}^{m-1})\mathbf{u}_{k-1}^m - B^T\mathbf{p}_{k-1}^m, \end{aligned} \tag{3.35a}$$

$$\mathbf{s}_{k-1} = \mathbf{g}((\mathbf{u}_{bc})^m) - B\mathbf{u}_{k-1}^m + C\mathbf{p}_{k-1}^m. \tag{3.35b}$$

Note that if we drop the $W^{(\rho)}$ term this gives an equivalent formulation of Picard linearisation. While Newton iteration can yield quadratic convergence, it only does so within a small region around the solution. Moreover, this region shrinks as the Reynolds number increases. Without a good initial guess, typically we might start with Picard iteration and swap to Newton iteration once sufficiently close to the (unknown) solution to gain quadratic convergence. Nonetheless, for smaller time-steps, the previous solution typically becomes a better initial guess. More details on the nonlinear iterative process and on finding appropriate solutions can be found in, for example, [80]. We now turn our attention to the generic linear systems to be solved.

3.5 The generic form of the linear systems

Following the discretisation, both spatial and temporal, and linearisation, the generic form of the Navier–Stokes system to be solved at each step is

$$\begin{pmatrix} F & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} \delta\mathbf{u} \\ \delta\mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{r} \\ \mathbf{s} \end{pmatrix}, \tag{3.36}$$

where upon Picard linearisation

$$F = \Delta t^{-1} M^{(\rho)} + N^{(\rho)} + A^{(\mu)} \tag{3.37}$$

while for Newton linearisation

$$F = \Delta t^{-1} M^{(\rho)} + N^{(\rho)} + W^{(\rho)} + A^{(\mu)}. \tag{3.38}$$

The coefficient matrix in (3.36) has a block structure known as a generalised saddle point form. We also need to solve the discrete advection system

$$\left(\Delta t^{-1} E + H\right) \mathbf{\Phi} = \mathbf{t}, \tag{3.39}$$

for the level set function, where $\mathbf{t}$ collects the knowns of this system.

To recap, the matrix $A^{(\mu)}$ represents the viscous term, $N^{(\rho)}$ and $W^{(\rho)}$ comprise the linearised approximation to the convective term, $M^{(\rho)}$ is a scaled mass matrix and B represents the negative divergence so that B^T is the gradient. The matrix C is a stabilisation term which is needed if an *inf–sup* unstable finite element pairing is chosen. Meanwhile, the matrix E represents the mass matrix corresponding to the level set function with the addition of streamline diffusion for stability. Further, H represents the relevant advective term in the level set equation, again with the addition of streamline diffusion for stability.

The primary difficulty and dominant computational cost is in solving the linearised Navier–Stokes system (3.36). Our work focuses on this key part of the overall two-phase flow solver algorithm. In practical applications the linear system (3.36) is large and sparse and hence is often solved using iterative Krylov methods, in particular, since we have a nonsymmetric system, we will consider GMRES [195]. However, the performance of such iterative methods depends very much on the conditioning of the problem. Thus, to have a practical and efficient solver one must utilise a good preconditioner. The next chapter outlines this need for scalable linear solvers for such incompressible flow problems and describes the key components involved. It further provides an overview of existing preconditioned iterative techniques before subsequent chapters discussed novel approaches aimed specifically at the two-phase problem.

Before moving on to considering solving linear systems, we first briefly describe the *inf–sup* condition required for the linearised Navier–Stokes part of the system to be stable and discuss the choice of finite elements used, including an example where an appropriate stabilisation matrix C is necessary.

3.6 The *inf–sup* condition

For mixed finite element methods an important criterion for well posedness of the problem and stable computation is the so-called *inf–sup* condition, also known as the Ladyzhenskaya–Babuška–Brezzi (LBB) condition. We detail first the abstract continuous setting, with a prototypical example being the Stokes equations, before discussing the required *discrete inf–sup* condition for the discretised system.

3.6.1 The *inf–sup* condition in infinite dimensions

We suppose that V and Q are two Hilbert spaces over the real numbers (these will be the velocity and pressure spaces). Let the corresponding norms be denoted $\|\cdot\|_V$ and $\|\cdot\|_Q$ respectively. Suppose that $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ and $b(\cdot,\cdot) : Q \times V \to \mathbb{R}$ are bounded bilinear functionals and $l_V(\cdot) : V \to \mathbb{R}$ and $l_Q(\cdot) : Q \to \mathbb{R}$ are bounded linear functionals. When considering the unique solvability of the variational problem: find $u \in V$ and $p \in Q$ such that

$$a(u,v) + b(p,v) = l_V(v), \tag{3.40a}$$

$$b(q,u) = l_Q(q), \tag{3.40b}$$

for all $v \in V$ and $q \in Q$, the *inf–sup* condition is that

$$\inf_{q \in Q\setminus\{0\}} \sup_{v \in V\setminus\{0\}} \frac{b(q,v)}{\|q\|_Q \|v\|_V} \geq c_b \tag{3.41}$$

holds for some constant $c_b > 0$. Defining the closed linear subspace $Z \subset V$ as

$$Z = \{v \in V : b(q,v) = 0 \ \forall q \in Q\}, \tag{3.42}$$

then if we further have coercivity of $a(\cdot,\cdot)$ on Z, that is

$$a(v,v) \geq c_a \|v\|_V^2 \ \forall v \in Z \tag{3.43}$$

for some constant $c_a > 0$, then there exists a unique $u \in V$ and $p \in Q$ solving the variational problem (3.40). Note that (3.43) is a sufficient but not necessary condition.

3.6.2 The discrete *inf–sup* condition

Upon discretisation of the abstract variational problem (3.40) it does not necessarily hold that the counterpart of the *inf–sup* condition (3.41) holds even if it does in the infinite dimensional case. As such, we require a discrete *inf–sup* condition to hold.

Consider discretisations depending on some mesh parameter $h \in (0,1)$ resulting in seeking solutions $u_h \in V_h \subset V$ and $p_h \in Q_h \subset Q$ such that

$$a(u_h,v_h) + b(p_h,v_h) = l_V(v_h), \tag{3.44a}$$

$$b(q_h,u_h) = l_Q(q_h), \tag{3.44b}$$

for all $v_h \in V_h$ and $q_h \in Q_h$. For unique solvability we must have that the following *discrete inf–sup* condition

$$\inf_{q_h \in Q_h\setminus\{0\}} \sup_{v_h \in V_h\setminus\{0\}} \frac{b(q_h,v_h)}{\|q_h\|_Q \|v_h\|_V} \geq c_d \tag{3.45}$$

holds for some constant $c_d > 0$, independent of h. Since this is not inherited from the infinite dimensional problem, this discrete *inf–sup* condition must be verified for each choice of finite element pairing. We similarly must have the discrete counterpart of the coercivity condition (3.43), this time on the kernel

$$Z_h = \{v_h \in V_h : b(q_h, v_h) = 0 \ \forall q_h \in Q_h\}, \tag{3.46}$$

requiring that there exists $c_c > 0$ such that

$$a(v_h, v_h) \geq c_c \|v_h\|_V^2 \ \forall v_h \in Z_h, \tag{3.47}$$

for a unique solution to the discrete variational problem (3.44). When considering the Stokes problem (with suitable boundary conditions) $a(\cdot, \cdot)$ is coercive on the whole of $V \subset \boldsymbol{H}^1$ so the coercivity condition does not contribute any difficulties for the discrete problem while the *inf–sup* condition does. The discrete *inf–sup* condition for the Stokes (or Navier–Stokes) problem reads as follows:

$$\min_{q_h \in Q_h \setminus \{0\}} \max_{\boldsymbol{v}_h \in V_h \setminus \{\boldsymbol{0}\}} \frac{|(q_h, \nabla \cdot \boldsymbol{v}_h)|}{\|q_h\|_{L^2} \|\boldsymbol{v}_h\|_{\boldsymbol{H}^1}} \geq \gamma, \tag{3.48}$$

for some constant $\gamma > 0$. It can be shown (see, for example, [52]) that the square of the *inf–sup* constant γ is precisely the minimum eigenvalue of the so-called Poincaré–Steklov operator, the continuous analogue of the Schur complement (see Chapter 4).

For the linearised Navier–Stokes equations, similar principles as discussed in this section apply, with the crux being the validity of the discrete *inf–sup* condition. We now discuss some example finite element pairings that we will use, including one which does not satisfy the discrete *inf–sup* condition and thus requires pressure stabilisation.

3.7 Finite element pairings for the Navier–Stokes problem and pressure stabilisation

In the majority of this work we will consider three choices for the velocity–pressure finite element spaces. These will be based on regular grids of square elements having side length h. We consider standard (nodal) Lagrange finite elements; these have piecewise polynomial basis functions. Perhaps the most well known pairings are the family of the Taylor Hood finite elements [107] which have the polynomial order for the velocity space being one higher than for the pressure space, this ensures the velocity space is rich enough in comparison to the pressure space that the discrete *inf–sup* condition (3.48) holds. We consider the lowest order member of the family on

quadrilaterals, known as the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ element pair, where the velocity space consists of continuous piecewise biquadratic polynomials and the pressure spaces comprises continuous piecewise bilinear polynomials.

As a second choice we consider equal order $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ elements. Here both velocity and pressure spaces use continuous piecewise bilinear polynomials. This element pairing is not naturally stable as it does not satisfy the discrete *inf–sup* condition and spurious pressure modes can result. As such a pressure stabilisation technique is needed when using this pairing; we follow the approach developed in [58, 25]. As noted in [65], the lack of stability of $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ approximation stems from the mismatch between the discrete divergence of the velocity field and the discrete pressure space $\boldsymbol{Q}_1$. The approach of [58] is to project the pressure into the appropriate space. This requires the L^2 projection from the pressure space into $\boldsymbol{P}_0$ (piecewise constant pressures), which we denote by Π_0. It is defined locally on each element T through a local averaging

$$\Pi_0 p_h|_T = \frac{1}{|T|}\int_T p_h \qquad \forall\ T \in \mathcal{T}_h, \tag{3.49}$$

where $\mathcal{T}_h$ is the set of $\boldsymbol{Q}_1$ elements used and $|T|$ denotes the area of element T. For the Navier–Stokes problem to ensure commensurate scaling the resulting stabilisation matrix is scaled by in the inverse of viscosity, as in [65]. With variable viscosity the appropriate stabilisation term is then given by

$$c(p_h, q_h) = \sum_{T\in\mathcal{T}_h}\int_T \mu^{-1}(p_h - \Pi_0 p_h)|_T(q_h - \Pi_0 q_h)|_T. \tag{3.50}$$

from which the associated stabilisation matrix C in (3.26e) can be constructed. This matrix can be assembled from element contributions in the standard manner.

For our third choice of finite element pairing considered we choose a discrete pressure space which is discontinuous (recall that p_h need only reside in L^2). For this we consider $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ elements, which are naturally *inf–sup* stable and conserve mass elementwise (for a constant density flow).[2] The velocity space comprises of continuous piecewise biquadratic polynomials while the pressure space is discontinuous (over element boundaries) and is piecewise linear; the notation of the minus sign in $\boldsymbol{P}_{-1}$ signifies that the polynomials are only piecewise continuous.

[2]Elementwise conservation of mass (for a constant density flow) is a consequence of piecewise constants living in the discontinuous pressure space. That is, the piecewise constant function which takes the value one on an element T and zero elsewhere is in $\boldsymbol{P}_{-1}$ and plugging this p_h in (3.20b) with $c(\cdot,\cdot) \equiv 0$ and using integration by parts gives $\int_{\partial T} \boldsymbol{u}_h \cdot \boldsymbol{n} = 0$. Thus, when the fluid has the same density, this yields conservation of mass.

There are a wide variety of other possible choices for the finite element spaces. This includes Raviart–Thomas elements [184, 154] and Brezzi–Douglas–Marini elements [37, 155], both of which are $H(div)$-conforming elements. That is, they are subspaces of

$$H(\mathrm{div}) = \left\{ u : \Omega \to \mathbb{R} \mid \nabla \cdot u \in L^2(\Omega) \right\}; \tag{3.51}$$

a larger space than H^1. Hence, additional work is required for their use; see [244, 243]. Further options include the Crouzeix–Raviart element [53] and the so-called mini element [6]. Discontinuous Galerkin methods can also be used for the Navier–Stokes equations [5, 49].

We now move to our main focus, solving the linear systems arising from the discrete variable coefficient Navier–Stokes equations (3.36).

Chapter 4

Preconditioning techniques for incompressible flow problems

This chapter focuses on solving the linear systems that arise in discretisations of incompressible flow problems, such as those modelled by the Stokes and Navier–Stokes equations. In particular, we consider iterative solution methods and the important requirement of good preconditioners. After addressing the need for preconditioning and reviewing appropriate general strategies for solving the (generalised) saddle point systems that arise, we ultimately concentrate on block preconditioners and Schur complement approximations. The following chapters detail our new work addressing preconditioning for two-phase flow.

It will be convenient to consider solving a generic (generalised) saddle point system of the form $K\mathbf{x} = \mathbf{b}$. Typically we will consider this system in the form most often used in incompressible fluid dynamics, namely

$$\begin{pmatrix} F & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix}, \tag{4.1}$$

for some velocity variables $\mathbf{u}$ and some pressure variables (Lagrange multipliers for the incompressibility constraint) $\mathbf{p}$. Note that we use the notation of F for the velocity block, as opposed to A. This is to recall that we are generally interested in solving systems where the velocity block is nonsymmetric; further A is reserved for Laplacian-type operators. More generally a generalised saddle point system need not have symmetry of the off-diagonal blocks B and B^T [17] and some of the methods we discuss are applicable in this case. Additionally some solution techniques allow $C = 0$, namely when an *inf–sup* stable discretisation is employed. In fact, some solver methodologies require $C = 0$; this shall be stated when reviewing such methods.

4.1 Iterative solution methods and the importance of preconditioners

Discretisation of PDEs often lead to large sparse linear systems that require efficient solvers. In practical scenarios, the large dimension of these systems typically prohibits the use of direct methods, or at least makes them slow and memory-intensive. As such, iterative methods are predominantly used to solve the systems to a required tolerance (and allow flexibility in the solution accuracy required). For sparse linear systems, such as those we consider, iterative Krylov approaches [194] are often favoured. These work by building up spaces (called Krylov subspaces) based on repeatedly applying the action of the coefficient matrix to a vector, the sparse nature of the matrix allowing the matrix–vector products to be relatively cheap. When solving $K\mathbf{x} = \mathbf{b}$ (without preconditioning) the appropriate Krylov subspace is

$$\mathcal{K}_k(K, \mathbf{b}) = \operatorname{span}\left\{\mathbf{b}, K\mathbf{b}, \ldots, K^{k-1}\mathbf{b}\right\}. \tag{4.2}$$

An important aspect of an iterative method is its convergence behaviour. It is well known that convergence of iterative Krylov methods depends on the conditioning of the coefficient matrix K. In particular, for symmetric matrices K the convergence of appropriate methods, such as the conjugate gradient (CG) [103] or minimum residual (MINRES) [169] method, is solely dictated by the eigenvalue spectrum of K along with right-hand side vector $\mathbf{b}$ (assuming exact arithmetic). Preconditioning is a way to obtain a more favourable spectrum of eigenvalues to gain faster convergence of the underlying iterative solver. For nonsymmetric problem (such as those arising from the Navier–Stokes problem) the issue is more complicated with eigenvalues alone not being descriptive of convergence. In practice, aiming for a well behaved spectrum (such as with few, clustered or bounded eigenvalues) can still be useful despite not being rigorously justified, in particular see [87]. The most commonly used Krylov iterative method for nonsymmetric problems is the generalised minimum residual (GMRES) method [195] (or one of its variants, for instance restarted GMRES [73] or GMRESR [237]). Rigorous bounds are possible (though typically pessimistic) for GMRES based on field-of-values analysis; see [213, 125, 141]. When preconditioning, symmetry (more properly the property of being self-adjoint) can be important to maintain if present in the linear system. However, since we consider nonsymmetric problems we do not pursue this (though note a nonsymmetric matrix can be self-adjoint in a nonstandard inner product; see [214] for example). As such, we consider the general preconditioning strategy of right-preconditioning.

A right-preconditioner for the linear system $K\mathbf{x} = \mathbf{b}$ is an invertible matrix P^{-1} used to solve the system by first solving $KP^{-1}\mathbf{y} = \mathbf{b}$ for $\mathbf{y}$ and then $P\mathbf{x} = \mathbf{y}$ for $\mathbf{x}$. The iterative method is performed on the preconditioned matrix KP^{-1} and so it is the properties of this matrix which define the convergence of the iterative method, however we also require the action of P^{-1} on vectors and so P must be chosen to ensure this action is relatively inexpensive. These two objectives are often at odds with each other. In fact, choosing $P = K$ would give the best properties for KP^{-1} but then solving $P\mathbf{x} = \mathbf{y}$ for $\mathbf{x}$ is our original problem so nothing has been gained. On the other hand, choosing $P = I$ gives the most easily computed action of P^{-1} but then our iterative method is applied to the original matrix and so, again, nothing has been gained. Finding a good preconditioner is often a non-trivial task and depends strongly on the particular linear system to be solved. While generic preconditioning strategies can be developed, often they are less potent than preconditioners specifically designed for a certain (small) class of problems. Indeed, the more structure of the underlying problem which can be utilised, the better a preconditioner will typically perform. Given no best preconditioner typically exists, especially for challenging problems of interest, there is a sense in which, as expressed in [247], preconditioning can be considered an art as opposed to a science. Nonetheless, mathematical investigation and careful analysis are indispensable tools for any would-be artist.

As well as the choice of preconditioner, we also have a choice of iterative method. For nonsymmetric problems, while GMRES is popular and has convergence which is monotonic, it has the unfortunate property that as the method progresses, each iteration becomes more expensive and requires more storage (there is no short-term recurrence as with CG or MINRES). Nonetheless, this instils the importance of using a good preconditioner to reduce the number of iterations required. Yet there are many other iterative methods designed for nonsymmetric problems which have a fixed work per iteration, including Bi-CGSTAB [236], QMR [77] and IDR [212]. However, necessarily these methods loose the optimality property of minimising the residual (in the Euclidean norm) and convergence can be more erratic with fewer guarantees and less well understood theory. The review article [211] contains more details on developments in Krylov subspace methods. In this work we focus on GMRES, which will be the iterative method used in our numerical experiments. Note that in large scale applications, due to the additional computational and storage costs required as the iteration progresses, often GMRES is restarted to keep a handle on the costs. However, after the first restart optimality is lost and allowing more iterations before restarting may not improve performance [73]. We do not consider restarted GMRES.

We detail the steps of the right-preconditioned GMRES method to solve $K\mathbf{x} = \mathbf{b}$ with preconditioner P; it will also provide a useful reference for a multipreconditioned approach that will be discussed in Chapter 8. We suppose the initial guess for GMRES is $\mathbf{x}_0 = \mathbf{0}$ so that the initial residual is $\mathbf{r}_0 = \mathbf{b}$ which, when normalised, provides the first basis vector in our Krylov subspace, $\mathbf{v}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|_2$. The basis is extended using the Arnoldi procedure and the optimal $\mathbf{x}$, minimising the residual norm, in the Krylov subspace is found using right-preconditioned GMRES as follows:

Algorithm 1 Right-preconditioned GMRES to solve $K\mathbf{x} = \mathbf{b}$ with preconditioner P using initial guess $\mathbf{x}_0 = \mathbf{0}$ so that $\mathbf{r}_0 = \mathbf{b}$ and $\mathbf{v}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|_2$.

1: **for** $k = 1, 2, \ldots$ **until** specified tolerance reached **do**
2: $\quad \mathbf{v}_{k+1} \leftarrow KP^{-1}\mathbf{v}_k$
3: $\quad$ **for** $j = 1, 2, \ldots, k$ **do** (Arnoldi using modified Gram–Schmidt)
4: $\quad\quad h_{j,k} \leftarrow \mathbf{v}_j^T \mathbf{v}_{k+1}$
5: $\quad\quad \mathbf{v}_{k+1} \leftarrow \mathbf{v}_{k+1} - h_{j,k}\mathbf{v}_j$
6: $\quad h_{k+1,k} \leftarrow \|\mathbf{v}_{k+1}\|_2$
7: $\quad \mathbf{v}_{k+1} \leftarrow \mathbf{v}_{k+1}/h_{k+1,k}$
8: $\quad$ **define** $\widehat{H}_k = \{h_{i,j}\}$ (an upper Hessenberg matrix)
9: $\quad$ **solve** $\mathbf{y} = \arg\min \left\| \|\mathbf{r}_0\|\mathbf{e}_1 - \widehat{H}_k\mathbf{y} \right\|_2$ where $\mathbf{e}_1 = [1, 0, \ldots, 0]^T$
10: For final iteration I **define** $V_I = [\mathbf{v}_1, \ldots, \mathbf{v}_I]$
11: $\mathbf{x}_I = \mathbf{x}_0 + P^{-1}V_I\mathbf{y}$

Note the linear least squares problem in line 9 of Algorithm 1 is solved via a QR factorisation of the upper Hessenberg matrix $\widehat{H}_k$ which can be updated from the previous QR factorisation of $\widehat{H}_{k-1}$ using just a single Givens rotation; see more details on the GMRES method in [194].

We note that, from the abstract functional analysis point of view, preconditioning can be seen as an essential component of iterative Krylov methods. In particular the matrix K may map outside of its domain and so re-applying K (in forming the Krylov subspace) has no meaning as the mappings between the spaces are not correct. As such, a preconditioner is required to map back to the original space so that forming the Krylov subspace makes sense. More background on this philosophy and in identifying so-called canonical preconditioners is found in the survey paper [148]; similar ideas on operator preconditioning are also discussed in [104]. Further, see the descriptions in [96] for more details on the CG and MINRES methods within the setting of continuous self-adjoint operators.

Before moving on we explain how we measure the convergence in both the linear and nonlinear iterations. When using GMRES to solve linear systems, in order to ascertain a suitable stopping criterion, we monitor the residual vector of the linear system given in (4.1). Suppose we are solving for the $k+1$th Picard iterate, then the linear residual is

$$\mathbf{R}^{(\ell)}_{\text{linear}} = \begin{pmatrix} \Delta t^{-1}M^{(\rho)} + N^{(\rho)}(\mathbf{w}^{(k)}) + A^{(\mu)} & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}^{(\ell)} \\ \mathbf{p}^{(\ell)} \end{pmatrix} - \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix}, \tag{4.3}$$

where $\mathbf{u}^{(\ell)}$ and $\mathbf{p}^{(\ell)}$ are the approximate solutions to the system after ℓ iterations of preconditioned GMRES. The notation of $N^{(\rho)}(\mathbf{w}^{(k)})$ is used here to recall that the convective part depends on the discrete velocity solution vector $\mathbf{w}^{(k)}$ at the previous Picard iteration. Denoting the corresponding discrete pressure solution vector at the kth Picard iterate by $\mathbf{q}^{(k)}$, we also define the nonlinear residual as

$$\mathbf{R}^{(k)}_{\text{nonlinear}} = \begin{pmatrix} \Delta t^{-1}M^{(\rho)} + N^{(\rho)}(\mathbf{w}^{(k)}) + A^{(\mu)} & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{w}^{(k)} \\ \mathbf{q}^{(k)} \end{pmatrix} - \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix}. \tag{4.4}$$

After the kth nonlinear iteration, we terminate the GMRES iteration for solving the linearised system given in (4.1) to give the $k+1$th nonlinear iterate once the relative residual norm decreases below a prescribed tolerance ε, that is we terminate once

$$\left\| \mathbf{R}^{(\ell)}_{\text{linear}} \right\|_2 \leq \varepsilon \left\| \mathbf{R}^{(k)}_{\text{nonlinear}} \right\|_2. \tag{4.5}$$

For each inner application of GMRES we will use an initial guess of the past solution $\mathbf{u}^{(0)} = \mathbf{w}^{(k)}$, $\mathbf{p}^{(0)} = \mathbf{q}^{(k)}$ and use a tolerance of $\varepsilon = 10^{-6}$ for termination in our results using IFISS. This method of tracking the convergence is the default used by IFISS. For Newton iteration the ideas are the same but now, in the velocity block F, as well as the term $N^{(\rho)}(\mathbf{w}^{(k)})$ we also include a second term $W^{(\rho)}(\mathbf{w}^{(k)})$ defined in (3.34).

In all our examples using IFISS, Picard iteration is terminated upon a relative reduction of 10^{-5} for the nonlinear residual. Meanwhile, our use of Newton iteration is terminated upon reaching a relative reduction of 10^{-10} for the nonlinear residual, except for results using $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ finite elements where a value of 10^{-8} is used instead.

We now return to the consideration of solvers for the linear systems that arise. For incompressible flow problems such as those modelled by the Stokes or Navier–Stokes equations, the system matrix has a (generalised) saddle point form. This natural block structure can be utilised when deriving efficient solution algorithms. One way is by considering block preconditioners, which will be our focus. However, many other strategies have also been considered and we first review some of these ideas.

4.2 Literature review of general strategies

Numerical solution of the Navier–Stokes equations is an extensively researched topic with a vast body of literature. There are a number of ways to formulate the equations as well as many possible discretisation, time-stepping and linearisation methods giving different linear systems to solve. We do not pursue a comprehensive review here and, instead, primarily provide an overview of some of the most common and promising work to solve formulations which result in (generalised) saddle point problems. Note, an excellent 2005 review on the numerical solution of saddle point systems is given by Benzi, Golub and Liesen [17]. A more concise overview of preconditioning techniques can be found in [24].

One class of methods for solving saddle point systems are null space methods [176, 187]. Early work on using such methods for incompressible flow problems can be found in [3, 97, 98, 197] under the name of the dual variable method. Null space methods utilise the null space of B (defining the constraint) to project down to solving a smaller system. However, the approach cannot be applied when $C \neq 0$, in our case requiring an *inf–sup* stable discretisation. The primary concern is finding such a null space, represented through a basis, and acquiring appropriate properties, such as sparsity of this basis or having a well conditioned basis. The method can also be viewed as a factorisation [187]. Further details and discussion can be found in [17, Chapter 6].

Techniques popular within the fluid dynamics literature are the Uzawa and inexact Uzawa methods. They are stationary schemes which requires a coupled iteration solving for both variables simultaneously. The Uzawa method is equivalent to a stationary Richardson iteration on the reduced Schur complement system. Typically the sub-solves required can be replaced by inexact solves with inexact variants being more efficient. In the symmetric case see, for example, [68, 256, 178] and, further, [35] in the nonsymmetric case. More recently, [205] provide approaches for generalised saddle point problems using variable relaxation parameters. For the steady Stokes problem, the Uzawa method converges at a rate independent of the mesh parameter h. However, poorer convergence is observed for the generalised Stokes case as well as linearisation of the Navier–Stokes equations such as in the Oseen problem, particularly with large Reynolds number flows. Nonetheless, preconditioning can be used to gain faster convergence of the Uzawa iteration [68]. Further details are also found in the review [17, Chapter 8]. The Uzawa method itself can also be considered as a block lower triangular preconditioner, highlighting its relationship to block preconditioners.

When seeking a preconditioner for the full saddle point system, one approach is to keep the saddle point structure but simplify the top left block F. These are known as constraint preconditioners since the constraints are retained in the form of the preconditioner. This still requires the solution of saddle point systems but the hope is that with the simplified block they are much easier to solve (and moreover do not need to be solved exactly). Such preconditioners are often derived and used through a factorisation. Examples and developments are found in [121, 60] for the symmetric case and [82, 28] for the nonsymmetric case applied to the Oseen problem. More recent work includes [216] where the so-called Schilders' factorisation [60] is extended to the nonsymmetric case. Recently, in [83], it has been shown that projected Krylov methods are mathematically equivalent to constraint preconditioned Krylov methods (with a well chosen initial guess) but require less memory. These techniques are also related to null space methods, indeed they use projections into the null space, but do not require a basis of B.

A different class of preconditioners, that have been well studied recently, are based on Hermitian and skew-Hermitian splitting (HSS). Initially HSS was introduced as a stationary iterative method in [10] but convergence is often slow, hence the approach is more commonly used in forming preconditioners [15, 16]. HSS-based preconditioners utilise a nonsymmetric form where in the bottom blocks of (4.1) the sign is reversed. Spectral properties of the HSS approach can be found in [210]. Numerical results in [19] demonstrate good performance when applied to the Navier–Stokes equations in rotation form, particularly for large Reynolds numbers and also in the unsteady case. Subsequently there have been various generalisations and extensions of these ideas, for example in [9, 14, 127]. There are a wide variety of other splitting type methods including those in [18, 20, 74, 75, 119] and further references within.

Multigrid methods are often used for sub-problems arising in the solution process, however, specialist multigrid techniques have been proposed for the full coupled saddle point systems arising in incompressible flow (as well as other applications). Early work includes [36, 241, 238, 252, 66, 32, 31, 174, 116]; more discussion can be found in the book [250] as well as the review [17]. A key component of multigrid is the smoother: these can either be coupled (often called Vanka smoothers [238, 146]) or decoupled [166] (also called segregated). Smoothers for saddle point problems are described under the framework of distributive relaxation schemes (or transforming iterations) [36, 252]. Decoupled smoothers can offer advantages in their efficiency [166]; effective methods can be based on distributed Gauss–Seidel relaxations [36],

incomplete LU factorisation (ILU) [252], inexact Uzawa methods [255], constraint-type preconditioning [32] as well as block factorisation and block preconditioning ideas we will discuss shortly [204, 174, 219, 245]. Algebraic multigrid (AMG) methods have also been proposed in [249, 185, 242]. A smoothed aggregation multigrid method for the Stokes problem is proposed in [113]. Some analysis of multigrid for the Stokes problem is given in [160, 102] and references within.

Another class of multi-level methods are domain decomposition methods [183], which are most often used as preconditioners for an iterative method. A discussion of methods designed specifically for saddle point problems is found in [225, Chapter 9]. Methods for the coupled Stokes problem include those based on overlapping Schwarz methods [124, 172], iterative substructuring methods [1, 173, 133], FETI methods [131, 132] and others [167]; see also [59]. Work on coupled domain decomposition methods for nonsymmetric saddle point systems, such as those arising from Navier–Stokes flow, has only recently began to be developed, such as in [168, 142, 86, 233, 203, 137, 110, 217]. See also [64, 54] for comparison with block preconditioners.

Perhaps the most common approach for solving the time-dependent problem is by using a projection method. Such methods go back to Chorin [46] and Temam [220]. These projection methods are based on the Helmholtz decomposition principle or Ladyzhenskaya theorem [129] and are often related to particular fractional time-stepping methods. An excellent overview of projection methods is given in [94]. Broadly speaking there are three classes of methods: pressure-correction schemes, velocity-correction schemes, and consistent splitting methods. In pressure-correction schemes a provisional velocity is solved for, that does not satisfy the incompressibility constraint, the pressure is then corrected by projecting this provisional velocity into a suitable space of divergence free functions. For velocity-correction schemes the roles are reversed, first the pressure is solved for and then, in a second step, the velocity is corrected. Consistent spitting methods are not truly projection methods but are similar to pressure-correction schemes as they first compute the velocity and then update the pressure. There is also a distinction between non-incremental schemes, which apply directly to the variables, and incremental schemes, which solve for increments of the variables. Recent work on projection methods includes [114, 181, 180, 95, 156, 130, 61, 57, 26, 62, 144]. They can also be used a preconditioners, as in [89]. Further, there are also some relations between projection methods and inexact algebraic factorisation schemes, such as the Yoshida method; see [175, 182, 94],

Another approach is to use incomplete LU factorisation (ILU) as a preconditioner. Here, specialist techniques must be used, such as those developed in the saddle point

ILU (SILU) preconditioner of [231] and the ILU preconditioner for nonsymmetric saddle point matrices of [126]. Earlier work applicable to the Navier–Stokes problem can be found in [47, 239, 251].

Yet more approaches include pressure correction schemes such as SIMPLE (semi-implicit method for pressure-linked equations) [171] and its variants [170, 232] and the Yoshida scheme [182] and its developments [196, 240, 186]. Further, augmented Lagrangian preconditioners [76, 21, 23] have proved promising. These methods can also be viewed as block preconditioners and so we discuss them further shortly. We now introduce the notion of block preconditioners.

4.3 Block preconditioning and Schur complements

Block preconditioning utilises block diagonal or block triangular preconditioners and typically stems from a block factorisation of the system. Such structure separates the different components (which may represent different physics, such as momentum conservation and mass conservation for incompressible flow), hence these ideas can be viewed as developing segregated solvers. For coupled physics problems such preconditioners can also be considered as physics-based preconditioners; here, known solvers can be used for some of the physics components required, for instance the convection–diffusion problem present in the F block. Block diagonal approaches are often used when maintaining symmetry is important (such as in the Stokes problem) so that iterative techniques such as MINRES can be applied (though see [33] for an alternative). Otherwise, block triangular preconditioners are favourable and are only marginally more expensive to apply.

Most commonly, block triangular preconditioners are based on the block LU-decomposition of K given by

$$K = \begin{pmatrix} F & B^T \\ B & -C \end{pmatrix} = \begin{pmatrix} I & 0 \\ BF^{-1} & I \end{pmatrix} \begin{pmatrix} F & B^T \\ 0 & -S \end{pmatrix} = LU, \tag{4.6}$$

where $S = BF^{-1}B^T + C$ is the (negative) Schur complement. Note this uses the pressure Schur complement (of F in K). If the block upper-triangular matrix U is used as a right-preconditioner then all eigenvalues of the preconditioned matrix $L = KU^{-1}$ are identically one. As shown in [153] for the case $C = 0$ and extended for non-zero C in [111], in this case GMRES would require only two iterations to compute the solution to the preconditioned system since the preconditioned system KU^{-1} has the minimum polynomial $(\lambda - 1)^2$.

Further block factorisation results in the form

$$K = \begin{pmatrix} F & B^T \\ B & -C \end{pmatrix} = \begin{pmatrix} I & 0 \\ BF^{-1} & I \end{pmatrix} \begin{pmatrix} F & 0 \\ 0 & -S \end{pmatrix} \begin{pmatrix} I & F^{-1}B^T \\ 0 & I \end{pmatrix} = \widetilde{L}\widetilde{D}\widetilde{U}. \quad (4.7)$$

Similarly to before, recombining the first two terms $\widetilde{L}\widetilde{D}$ we can obtain a block lower-triangular preconditioner. Using this as a right-preconditioner for K we see that the preconditioned system is a similar matrix to $\widetilde{U}$ (with the change of basis matrix being $\widetilde{L}\widetilde{D}$) and so again has a minimum polynomial $(\lambda - 1)^2$. If F is symmetric positive definite as in the Stokes problem, so that K is symmetric indefinite, the (negative) Schur complement is also positive definite. Then, in order to have a positive definite preconditioner (unlike $\widetilde{D}$ in (4.7)), we can use a preconditioner of the form

$$P_D = \begin{pmatrix} F & 0 \\ 0 & S \end{pmatrix}. \quad (4.8)$$

Now, in the case when $C = 0$, the right-preconditioned matrix is given by

$$KP_D^{-1} = \begin{pmatrix} I & -B^T S^{-1} \\ BF^{-1} & 0 \end{pmatrix} \quad (4.9)$$

and the minimum polynomial is $(\lambda - 1)(\lambda^2 - \lambda - 1)$, having three distinct roots; as such MINRES would only require three iterations to compute the solution. Note that in order to maintain symmetry, MINRES uses symmetric preconditioning not right-preconditioning but a simple similarity transformation argument shows that we need only consider (4.9). See [56] for details on block diagonal preconditioners for nonsymmetric indefinite systems.

Application of both the block diagonal and block triangular preconditioners coming from such exact factorisation requires solutions of linear systems involving F and S. That is, require the action of the inverses F^{-1} and S^{-1}. For large systems, these actions are not easily applied and S is generally a dense matrix despite K being sparse. As discussed in [17], using these preconditioners exactly results in an algorithm which is essentially as expensive as computing the inverse of K directly using

$$K^{-1} = \begin{pmatrix} F^{-1} - F^{-1}B^T S^{-1} BF^{-1} & F^{-1}B^T S^{-1} \\ S^{-1}BF^{-1} & -S^{-1} \end{pmatrix}. \quad (4.10)$$

Instead, the results of [153, 111] suggest using approximations of F and S in the block preconditioner. The potency of the resulting preconditioners are then determined by the quality of the approximations used. As such, we consider block (upper) triangular preconditioners of the form

$$P = \begin{pmatrix} \widehat{F} & B^T \\ 0 & -\widehat{S} \end{pmatrix}, \quad (4.11)$$

where $\widehat{F}$ and $\widehat{S}$ approximate the velocity block F and the (negative) Schur complement S respectively. The inverse of the preconditioner P is given by

$$P^{-1} = \begin{pmatrix} \widehat{F}^{-1} & \widehat{F}^{-1}B^T\widehat{S}^{-1} \\ 0 & -\widehat{S}^{-1} \end{pmatrix}, \tag{4.12}$$

so that its action on $\mathbf{v} = (\mathbf{v}_u, \mathbf{v}_p)^T$ in Algorithm 1, yielding $\widehat{\mathbf{v}} = (\widehat{\mathbf{v}}_u, \widehat{\mathbf{v}}_p)^T = P^{-1}\mathbf{v}$, can be given componentwise as

$$\widehat{\mathbf{v}}_p = -\widehat{S}^{-1}\mathbf{v}_p, \tag{4.13a}$$

$$\widehat{\mathbf{v}}_u = \widehat{F}^{-1}\left(\mathbf{v}_u - B^T\mathbf{v}_p\right). \tag{4.13b}$$

Thus the preconditioner needs a single application of both $\widehat{F}^{-1}$ and $\widehat{S}^{-1}$. The same is true for block diagonal preconditioners.

In the case of Navier–Stokes flow, F is a convection–diffusion type operator for which known methods for approximate solution can be applied, for example multigrid techniques [71, Chapter 7]. Within multigrid one has a sequence of grids: from the fine grid, where we wish to compute the solution, down to a coarser grid, where an exact or approximate solution is much cheaper. Multigrid is an iterative processes which aims to use the hierarchy of grids to efficiently reduce the iteration error. Essential components are the grid transfer operators, which allow us to traverse up and down the sequence of grid, and smoothers which aim to reduce the iteration error on a particular grid (by eliminating the oscillatory parts of the error, hence *smoothing* the error) so that it is well represented on a coarser grid. When using multigrid for convection-diffusion problems it is important to have a good choice of discretisation; in particular, features of the problem may not be well resolved on the coarser grid and this be taken account within multigrid, for instance by using a streamline diffusion discretisation. Furthermore, the choice of smoother employed is key and for convection–diffusion problems must take account of the flow. If the general flow direction is known, then a smoother which sweeps along the direction of the flow is appropriate, such as a step of an appropriately ordered (block) Gauss–Seidel iteration. For multi-directional or rotating flows, one can use a multi-directional smoother which uses sweeps in different directions; see [71, Chapter 7]. Finally, when a stable coarse grid discretisation is used, standard grid transfer operators can work well, at least in the single-phase case. Nonetheless, specialist operator-dependent grid transfer methods which take account consideration of the underlying flow are possible, see [226, Section 7.7], and can be advantageous for problems with discontinuous coefficients such as those arising in two-phase flow.

Typically the main challenge in applications of block preconditioners is an efficient yet faithful approximation to the Schur complement S. It is the approximation $\widehat{S}$, also called the Schur complement preconditioner, that will be the focus of our work. Nonetheless, we note that for two-phase flow, a good approximation of F can still be a challenge; in our experience, the methods available in IFISS [209] can be poor.

More generally, block preconditioners can incorporate full factorisations, such as (4.7), where certain terms are approximated. In particular, [64] consider approximate block factorisation (ABF) methods (see also [182]) which utilise

$$K = \begin{pmatrix} F & B^T \\ B & -C \end{pmatrix} \approx \begin{pmatrix} I & 0 \\ BH_1 & I \end{pmatrix} \begin{pmatrix} F & 0 \\ 0 & -\widehat{S} \end{pmatrix} \begin{pmatrix} I & H_2B^T \\ 0 & I \end{pmatrix} = \widetilde{K}, \tag{4.14}$$

where H_1 and H_2 are potentially different approximations to F^{-1} (note that [64] also do not require symmetry of the off-diagonal B blocks like we have here). Again, the approximations used define different preconditioners and the overall performance depends on the quality of these approximations. The error in the approximate block factorisation is given by

$$K - \widetilde{K} = \begin{pmatrix} 0 & B^T - FH_2B^T \\ B - BH_1F & \widehat{S} - \big(BH_1FH_2B^T + C\big) \end{pmatrix}. \tag{4.15}$$

Note that the case $H_2 = F^{-1}$ gives a "momentum conserving" approach since the momentum equation is unperturbed by the approximate factorisation. On the other hand, if $H_1 = F^{-1}$ and $\widehat{S} = BH_2B^T + C$ then the approach is "mass conserving" in the sense that the discrete divergence-free constraint remains unmodified.

4.4 Literature review of block preconditioners

In this section we review block preconditioners devised for incompressible fluid flow problems. As discussed above, the primary difference comes in the approximation of the Schur complement. In general we will be most interested in the Navier–Stokes case and, moreover, two-phase flow. However, we first discuss the bulk of the literature which is aimed at a single fluid and initially discuss appropriate Schur complement approximations for the Stokes problem.

For steady-state Stokes flow, where F consists solely of a diffusion term, the Schur complement is spectrally equivalent to the pressure mass matrix M_p, which itself is spectrally equivalent to its diagonal $\widehat{M_p} = \mathrm{diag}(M_p)$; see [246]. Thus $\widehat{M_p}$ can be used as a good approximation to S in (4.8) with preconditioned MINRES. Alternatively the action of M_p^{-1} is well approximated through application of a small number of

Chebyshev semi-iterations [248]. In the generalised Stokes problem (time-dependent case) there is now a balance between the time-stepping term and viscous diffusion term in F. In this case the well known Cahouet–Chabard preconditioner can be used [42, 34] where the inverse Schur complement is approximated through an appropriate balance of the mass term M_p and a pressure space Laplacian term $BM^{-1}B^T$ given by

$$\widehat{S}^{-1} = \nu M_p^{-1} + \Delta t^{-1}(BM^{-1}B^T)^{-1}, \tag{4.16}$$

where M is the velocity mass matrix. We will discuss this approach a little more shortly in Section 4.5 and now move to consider Navier–Stokes flow. While methods from the Stokes case can be applied for Navier–Stokes flow and, for instance, using the mass matrix as an approximation to the Schur complement in the stationary case results in mesh-independent convergence (see, for example, the analysis in [128]) their performance significantly deteriorates as convection becomes more dominant. As such, more sophisticated techniques are required.

A taxonomy of approximate block factorisation (ABF) methods for incompressible Navier–Stokes flow is given in [64]. Though not originally envisaged within this framework, this includes SIMPLE [171] and other SIMPLE-type methods [232]. For instance, the SIMPLE method uses (4.14) with $H_1 = F^{-1}$, $H_2 = \mathrm{diag}(F)^{-1}$ (the diagonal matrix consisting of the inverse of the diagonal entries of F), along with $\widehat{S} = B\mathrm{diag}(F)^{-1}B^T + C$. Thus, when considered as a preconditioner for a Krylov method, the preconditioner is given by

$$P_{\text{SIMPLE}} = \begin{pmatrix} F & 0 \\ B & -(B\mathrm{diag}(F)^{-1}B^T + C) \end{pmatrix} \begin{pmatrix} I & \mathrm{diag}(F)^{-1}B^T \\ 0 & I \end{pmatrix}. \tag{4.17}$$

This approximation is "mass conserving", that is, only modifying the momentum equation. Note an additional damping parameter can be introduced too; see [64]. A SIMPLE-like block upper-triangle preconditioner is considered in [71] and is given by

$$P_{\text{SIMPLE-like}} = \begin{pmatrix} F & B^T \\ 0 & -(B\mathrm{diag}(F)^{-1}B^T + C) \end{pmatrix}. \tag{4.18}$$

We shall give comparisons using this block upper-triangle preconditioner (4.18) in our numerical results. The potency of SIMPLE depends on how well the diagonal of F approximates F. In the time-dependent case with small time-steps the mass term in F (see (3.37)–(3.38)) is dominant and the approximation is good. Otherwise, however, SIMPLE can perform poorly, especially in the steady case. In addition to SIMPLE, there are a number of SIMPLE-type methods which also use diagonal approximations of F, as discussed in [64].

Another approach in the time-dependent case is the Yoshida method [182]. This amounts to using $H_1 = H_2 = F^{-1}$ and $\widehat{S} = \Delta t B M^{-1} B^T + C$ in (4.14) and hence the preconditioner

$$P_{\text{Yoshida}} = \begin{pmatrix} F & 0 \\ B & -(\Delta t B M^{-1} B^T + C) \end{pmatrix} \begin{pmatrix} I & F^{-1} B^T \\ 0 & I \end{pmatrix}. \tag{4.19}$$

In practice the velocity mass matrix M is replaced by a diagonal approximation. The Yoshida scheme is "momentum conserving". Note that if Δt is very small the dominant term in F is the time-stepping term $\Delta t^{-1} M$ and hence the Yoshida scheme corresponds to neglecting all other terms of F in the Schur complement. As with SIMPLE, performance is typically only good when sufficiently small time-steps are taken. Extensions of the Yoshida scheme for higher order accuracy in time are given in [196, 240, 186], some of which use a more general approximate factorisation than (4.14).

Other block preconditioners for Navier–Stokes flow are based on approximate commutators, primarily these are the pressure convection–diffusion (PCD) and least-squares commutator (LSC) preconditioners, which are discussed in detail in [71]. Though most often described through an approximate commutation relationship, the PCD approach was originally devised in [118] by considering Green's tensors. Further developments of this approach are found in [207, 72, 151, 71] while [138, 139, 69, 141, 165] provide some analysis; see also [140]. The PCD preconditioner requires a convection–diffusion operator projected onto the discrete pressure space and this operator must be constructed specifically for the preconditioner. In search of a more automatic approach, the LSC preconditioner described in [63], but developed from [67], is based solely on algebraic considerations of minimising the norm of the commutator in a least-squares sense; see also [78]. Subsequent developments are given in [65, 72, 71] while eigenvalue bounds are found in [69, 165]. Both of these commutator-based preconditioners show scalable results in the tests of [71] and [54]. We note that, along with a comparison of PCD and LSC, another preconditioner motived by commutation is given in [165].

An alternative idea is to augment the linear system first so that the augmented system admits a simpler approximation to its Schur complement. These ideas feature in the augmented Lagrangian preconditioners devised in [21, 23]. Assuming for now that $C = 0$, using the augmented Lagrangian method [76] we replace the original saddle point system (4.1) with

$$\begin{pmatrix} F + \gamma B^T D^{-1} B & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{f} + \gamma B^T D^{-1} \mathbf{g} \\ \mathbf{g} \end{pmatrix}, \tag{4.20}$$

for some symmetric positive definite matrix D and parameter $\gamma > 0$. Note that, by construction, the solutions $\mathbf{u}$ and $\mathbf{p}$ remain unchanged. Further, if D is diagonal or block diagonal with small blocks then $F_\gamma = F + \gamma B^T D^{-1} B$ will be sparse. A block upper-triangular preconditioner is then used on the augmented system (4.20) of the form

$$P_\gamma = \begin{pmatrix} \widehat{F}_\gamma & B^T \\ 0 & -\widehat{S}_\gamma \end{pmatrix}, \tag{4.21}$$

where, similarly to before, $\widehat{F}_\gamma$ approximates F_γ and $\widehat{S}_\gamma$ approximates $S_\gamma = B F_\gamma^{-1} B^T$. In the steady-state case, under a suitable choice for γ (often $\gamma = 1$ in practice), [21] use the Schur complement approximation, defined through its inverse,

$$\widehat{S}^{-1} = \mu \widehat{M}_p^{-1} + \gamma D^{-1}. \tag{4.22}$$

In practice the positive definite matrix D used is a diagonal approximation to the mass matrix, in particular $D = \widehat{M}_p$ allows the terms in (4.22) to be combined. In [23] the time-dependent case is also considered and now the appropriate inverse Schur complement approximation includes an additional term in the same manner as the Cahouet–Chabard preconditioner. While the Augmented Lagrangian approach allows for a relatively simple approximation to the Schur complement, the modified F block becomes a more difficult task, however a specialist multigrid method to yield the action of F_γ is developed in [21] based on ideas in [201]. Further, modified augmented Lagrangian preconditioners are suggested in [23] where the submatrix $\widehat{F}_\gamma$ is block triangular (for blocks depending on the different components of the velocity) and now involves approximately inverting scalar convection–diffusion operators, thus more standard multigrid methods can be applied. This can reduce the overall solution time required. Extension to the case of $C \neq 0$ is also given in [23]. Numerical results for these Augmented Lagrangian preconditioners show mesh-independent convergence and only mild dependence on the Reynolds number, in particular when the Reynolds number becomes large. Rigorous analysis based on field-of-values bounds that prove h-independent convergence are derived in [22], with a suitably chosen γ. In the ideal case, where the velocity submatrix is solved exactly, the convergence rate of the original augmented Lagrangian preconditioner is also proved to be independent of the viscosity parameter and the time-step.

We note that effective block preconditioners have also been devised for the Navier–Stokes equation in rotation form in [158, 159] (see also [143, 162]); here an elliptic problem is solved to provide a Schur complement approximation. These methods

perform particularly well for small viscosity parameters (high Reynolds numbers) and are quite robust to the mesh parameter. A HSS preconditioner is also considered for the rotation form of the equations in [19] and again performs particularly well for small viscosity parameter.

The preconditioners we have described above are primarily considered in the case of constant density and viscosity flows and, to the author's knowledge, little has been explored for preconditioning variable density and viscosity Navier–Stokes problems. However, recently, augmented Lagrangian preconditioners [21, 23] were applied to the variable viscosity case in [100] and also to incompressible non-Newtonian flows in [101]; see also [7] for a simpler approach incorporating variable density in the time-dependent case.

In the case of variable coefficient problems, such as two-phase flow, most work is devoted to the Stokes problem. Here, a pressure mass matrix inversely scaled by the viscosity, denoted $M_p^{(1/\nu)}$, is a good choice of Schur complement preconditioner for the stationary problem, as shown in the case of two-phase flow in [164] and more generally investigated for variable viscosity in [90]; see also [41, 149].[1] By considering an abstract parameter dependent saddle point system, this is extended to two-phase non-stationary Stokes flow in [161] and can be seen as generalising the Cahouet–Chabard preconditioner [42, 34]. Similar block preconditioners are constructed in [43] for variable density and viscosity Stokes problems when using finite volumes applied on uniform staggered grids, showing promising results. This work brings together finite element literature on solving the coupled velocity–pressure system making use of Schur complements and applies such techniques within the finite volume setting, where the dominant paradigm has been splitting or projection methods [94]. A variant of LSC which takes into account viscosity contrast is given in [150] and is used for their studies in computational geodynamics. Another approach is the Schur method described in [230]; this paper also examines what is effectively the same LSC variant as [150]. Recently, a more favourable LSC-type method for the variable viscosity Stokes problem is presented in [190] and called weighted-BFBT based on the name of the earlier method given in [67]. While these methods hold potential to be used for variable coefficient Navier–Stokes flow we have found no reference to such application. As such, we consider the adaptation of such block preconditioners to two-phase Navier–Stokes flow, in particular the PCD as well as LSC approaches.

[1] Note that, as expected, this choice of appropriately scaled mass matrix $M_p^{(1/\nu)}$ (diagonalised if necessary) is found favourable as the choice of D in the augmented Lagrangian preconditioners when applied to variable viscosity problems.

4.5 Motivation for approximate commutators and the Cahouet–Chabard preconditioner

As emphasised above, for block preconditioners a crucial ingredient is a cheap but faithful approximation to the Schur complement. Several effective techniques are based on approximate commutators: we give some motivation for such ideas. For simplicity, consider the case $C = 0$ then the (negative) Schur complement is given by $S = BF^{-1}B^T$. Now consider the continuous operators in the constant viscosity Stokes case, the analogue to the Schur complement is the so-called Poincaré–Steklov operator $\mathcal{S} : L^2 \to L^2$ given by

$$\mathcal{S} = \nabla \cdot (\nu \nabla \cdot \nabla)^{-1} \nabla . \tag{4.23}$$

Heuristically, and neglecting boundary conditions, we expect this operator to be close to the (scaled) identity operator $\nu^{-1}\mathcal{I}$ (see, for example, [145]). This correspondence is proved rigorously in [208]. In the finite element context the identity operator becomes a mass matrix and so the corresponding inverse Schur complement approximation is $S^{-1} \approx \nu M_p^{-1}$. Note the same conclusion is reached if we commute either the divergence or the gradient operator with this inverse Laplacian term, which then cancels. In the time-dependent Stokes case, with time-step Δt, we have

$$\mathcal{S} = \nabla \cdot \left(\nu \nabla \cdot \nabla + \Delta t^{-1} \mathcal{I}\right)^{-1} \nabla . \tag{4.24}$$

Again using a commutation argument or other heuristic we reach the conclusion that the inverse operator is approximately

$$\mathcal{S}^{-1} \approx (\nabla \cdot \nabla)^{-1} \left(\nu \nabla \cdot \nabla + \Delta t^{-1} \mathcal{I}\right) = \nu \mathcal{I} + \Delta t^{-1} (\nabla \cdot \nabla)^{-1} . \tag{4.25}$$

We recognise this as the continuous analogue of the Cahouet–Chabard preconditioner (4.16). The more rigorous arguments of Cahouet and Chabard [42] utilise Fourier analysis. We note that, with periodic boundary conditions, in Fourier space the operators all commute, as such this gives a more formal use of commutation. Related concepts are given in [140] which considers pseudo-differential operators for Schur complement preconditioning.

When considering the discretised case, commuting operators become commuting matrices for a Schur complement approximation. However, in practice it is highly unlikely that these matrices commute exactly, though a special case is given in [72]

where the matrices do commute (with some assumptions). When aiming to commute with the divergence we have

$$\text{“}BF^{-1} \approx F_p^{-1}B\text{”}, \tag{4.26}$$

where now it is clear from the dimensions of the matrices that the new matrix F_p acts on the pressure space; note B is a rectangular matrix. Formal manipulation yields

$$\text{“}BF \approx F_pB\text{”}, \tag{4.27}$$

an expression which now only requires the forward operators and no inverses. The choice of matrix F_p is motivated in order to, in some appropriate sense, allow this commutation to hold approximately. The notion of commuting the divergence with the convection–diffusion operator is the approximate commutation behind which the approximate commutator methods discussed above are based. Note that in the finite element case, appropriate mass matrices are required for discrete commutation as opposed to simply (4.27). We now consider some approximate commutator based preconditioners for the Navier–Stokes problem.

Chapter 5

Pressure convection–diffusion preconditioning and extension using weighted commutators

This chapter details some of our main work on preconditioning the variable coefficient Navier–Stokes equations stemming from two-phase flow. In particular, this chapter focuses on the ideas behind pressure convection–diffusion (PCD) preconditioning. We first present the underlying ideas in the single-phase case and note a relation with the Cahouet–Chabard preconditioner. This viewpoint then affords us a mechanism to understand how a two-phase version of PCD should be formulated by analogy to an appropriate two-phase generalisation of the Cahouet–Chabard preconditioner. It further shows that a new form for PCD is necessary in general. However, this viewpoint still leaves open questions on the precise specifications and the important consideration of boundary conditions on the operators involved.

In order to fully describe the formulation of a two-phase PCD preconditioner we develop ideas based on novel weighted commutators. These arguments explain the necessary scaling required within two-phase PCD and allow us to gain a handle on the prescription of appropriate boundary conditions for the constructed pressure space operators that are required. Illustrative numerical results verifying our new approach will be given and we will demonstrate the effectiveness of the preconditioner and how, in particular for inflow–outflow problems, our choice of boundary conditions provides scalable performance.

5.1 The single-phase PCD preconditioner

5.1.1 Derivation of the PCD preconditioner

We first describe the formulation of the original PCD preconditioner, for a single-phase flow where the fluid has constant density and viscosity. While also being derived using Green's tensors [118], we present approximate commutator arguments [207] which lead to the formulation of an approximate Schur complement operator. A good introduction to these ideas is found in the book [71]. We note that the approach is derived in the case of the Oseen linearisation (3.32) and the same approximation is used for Newton linearisation (3.33); see [69].

The starting point lies in an approximate commutation between the divergence[1] operator $\mathcal{B}$ and convection–diffusion operator $\mathcal{F}$. Suppose we can find an operator $\mathcal{F}_p$ such that (formally) the commutator

$$\mathcal{E} = \mathcal{B}\mathcal{F} - \mathcal{F}_p\mathcal{B} \tag{5.1}$$

is small (in some appropriate sense). Note that $\mathcal{F}_p$ is an operator which acts on the pressure space as opposed to the velocity space. Using the approximation $\mathcal{E} \approx 0$, a formal manipulation of (5.1), namely applying $\mathcal{F}^{-1}\mathcal{B}^T$ from the right and $\mathcal{F}_p^{-1}$ from the left, results in the Schur complement approximation

$$\mathcal{B}\mathcal{F}^{-1}\mathcal{B}^T \approx \mathcal{F}_p^{-1}\mathcal{B}\mathcal{B}^T. \tag{5.2}$$

Here we use $\mathcal{B}^T$ to denote the adjoint[2] operator to $\mathcal{B}$, namely the gradient operator, to maintain an analogue with the discrete case. In application of a block preconditioner we require the action of the inverse of the Schur complement; in the continuous case this is then

$$\left(\mathcal{B}\mathcal{F}^{-1}\mathcal{B}^T\right)^{-1} \approx (\mathcal{B}\mathcal{B}^T)^{-1}\mathcal{F}_p. \tag{5.3}$$

In this approximation we have a Laplacian-type term $\mathcal{B}\mathcal{B}^T$ to invert but crucially we only require the application of $\mathcal{F}_p$, which should contain information about the non-normality inherent in the Schur complement, as opposed to inverting such an operator.

[1] In the spirit of (4.27) we might consider the *negative* divergence operator since this is what the matrix B represents. However, for convenience and clarity, we drop the negative sign and consider $\mathcal{B} = \nabla\cdot$ since it does not affect any of the arguments. In particular, note that the transformation $B \to -B$ does not change the Schur complement $S = BF^{-1}B^T + C$.

[2] Note that, since $\mathcal{B}$ represents the divergence, the adjoint should be the *negative* gradient. Again, for convenience and clarity we will drop the negative sign; see footnote 1.

The crucial unknown operator required in the commutator-based approximation of the (inverse) Schur complement is $\mathcal{F}_p$. In the PCD methodology this operator is derived from considering the continuous operators in (5.1) and making a (restrictive) assumption on the advective term in $\mathcal{F}$. As such, these arguments are somewhat heuristic in nature when applied to the general form of advective term present in $\mathcal{F}$. In principle, such arguments could be extended to other saddle point problems with the blocks representing other operators, however, we are not aware of any applications where this has been done other than for incompressible flow problems. Nonetheless, there are similarities with work in [140] on Schur complement preconditioning using the Schur complement of the symbol (which utilises the Fourier transform) of the operators. The natural setting of this approach requires pseudo-differential operators. In [140] the methodology is applied to equations describing magnetostatics as well as the Darcy, Stokes and Oseen problems.

Let us consider the (constant coefficient) stationary case of the Oseen problem. Here we have that $\mathcal{F}$ takes the form

$$\mathcal{F} = -\mu\,(\nabla\cdot\nabla\,) + \rho\,\boldsymbol{w}\cdot\nabla\,, \tag{5.4}$$

where μ and ρ are constants and the wind vector $\boldsymbol{w}$ corresponds to the previous approximation of the velocity $\boldsymbol{u}$ in the Picard iteration. We seek an operator $\mathcal{F}_p$ so that the commutator (5.1) vanishes as far as possible. In general we cannot fulfil that it vanishes everywhere and instead we must follow a more heuristic argument. To this end we suppose in the arguments that follow that the wind $\boldsymbol{w}$ is constant in space. Then we have that

$$\mathcal{B}\mathcal{F}\boldsymbol{u} = \nabla\cdot(-\mu\,\nabla\cdot\nabla\,\boldsymbol{u} + \rho\,\boldsymbol{w}\cdot\nabla\,\boldsymbol{u}) = (-\mu\,\nabla\cdot\nabla\, + \rho\,\boldsymbol{w}\cdot\nabla\,)\,\nabla\cdot\boldsymbol{u} = \mathcal{F}_p\mathcal{B}\boldsymbol{u}, \tag{5.5}$$

where $\mathcal{F}_p$ is a scalar version of the vector operator $\mathcal{F}$. Thus, neglecting boundary conditions for now, $\mathcal{E}$ is zero; indeed, it is noted in [71] that $\mathcal{E}$ would vanish if $\boldsymbol{w}$ were a constant and the operators were defined on an unbounded domain.

In the time-dependent case, $\mathcal{F}$ includes an additional term which is a scaling of the identity, namely $\mathcal{T} = \Delta t^{-1}\rho$. Hence this additional term commutes with the divergence $\mathcal{B}$ without issue, namely

$$\mathcal{B}\mathcal{T}\boldsymbol{u} = \nabla\cdot\left(\Delta t^{-1}\rho\,\boldsymbol{u}\right) = \Delta t^{-1}\rho\,\nabla\cdot\boldsymbol{u} = \mathcal{T}_p\mathcal{B}\boldsymbol{u}. \tag{5.6}$$

Thus, we need only add the equivalent scalar term $\mathcal{T}_p = \Delta t^{-1}\rho$ to $\mathcal{F}_p$ when required to give, in general, the pressure space operator

$$\mathcal{F}_p = -\mu\,(\nabla\cdot\nabla\,) + \rho\,\boldsymbol{w}\cdot\nabla\, + \Delta t^{-1}\rho. \tag{5.7}$$

To derive the PCD preconditioner from the commutator we must discretise (5.3) to give an approximation of the inverse of the Schur complement which, supposing for now that $C = 0$, is given by $S^{-1} = (BF^{-1}B^T)^{-1}$. In our setting of a mixed finite element formulation, we let M_p be the mass matrix on the pressure space. Then the discretisation of (5.3) becomes

$$\left(M_p^{-1}BF^{-1}B^T\right)^{-1} \approx A_p^{-1}F_p, \tag{5.8}$$

where A_p represents a discrete Laplacian (stemming from $\mathcal{B}\mathcal{B}^T$) and F_p is a discrete form of $\mathcal{F}_p$. Note that, when forming the discrete counterpart of the continuous operators, mass matrices are required to give appropriate scaling; see [71, Problem 9.4]. From (5.8) we obtain the pressure convection–diffusion (PCD) approximation to the inverse Schur complement

$$\widehat{S}^{-1} = A_p^{-1}F_pM_p^{-1}. \tag{5.9}$$

Remark 4 (Derivation of the PCD approach) *We note that our derivation of PCD uses a subtly different exposition from those given in the literature since we discretise only at the very end, namely we discretise the inverse Schur complement approximation* (5.3) *with* $\mathcal{F}_p$ *given by* (5.7) *and not the commutator* (5.1)*; as such, there is no discrete commutator considered. While this does not change the ultimate outcome, it allows us to directly discretise the Laplacian term* $\mathcal{B}\mathcal{B}^T$ *in a manner most appropriate to the discretisation used. Further, it upholds our viewpoint that PCD stems from considering approximately commuting operators as opposed to matrices. That is, the manipulation is best thought of in the continuous setting rather than the discrete setting and that the Schur complement preconditioner* (5.9) *approximates an appropriate infinite-dimensional (pseudo-differential) operator (see also ideas in [140]). Moreover, the operators considered come equipped with boundary conditions which allows us to analyse and understand what appropriate boundary conditions are for the new pressure space operators required, something which is, at best, unclear when simply considering the matrices.*

Remark 5 (Choice of commutator) *We remark that the commutator can also be taken with the gradient instead of the divergence, as originally done in [207]. The effect of this is that the ordering of the operators in* (5.9)*, our approximate inverse Schur complement* $\widehat{S}^{-1}$*, is reversed. It was seen when considering boundary conditions that the choice of the divergence was favourable; see [71, Remark 9.3]. See also [151] for a study on ordering of the operators in PCD.*

The matrices required in the PCD inverse Schur complement approximation (5.9) depend on the discretisation used. For simplicity, assume for now that the pressure approximation is continuous, though extensions can be made when a discontinuous pressure approximation is used (see [71]); we will discuss more in Section 5.4.2. In this case, with pressure basis functions $\{\psi_j\}$, we can define A_p to be the sparse pressure Laplacian

$$A_{p;i,j} = \int_\Omega \nabla\,\psi_j \cdot \nabla\,\psi_i. \tag{5.10}$$

Further, discretisation of the pressure space convection–diffusion operator (5.7) yields

$$F_{p;i,j} = \mu \int_\Omega \nabla\,\psi_j \cdot \nabla\,\psi_i + \rho \int_\Omega (\boldsymbol{w} \cdot \nabla\,\psi_j)\,\psi_i + \Delta t^{-1} \rho \int_\Omega \psi_j\,\psi_i. \tag{5.11}$$

Note that this can be written as the sum of the diffusion part plus the convection part plus a time-stepping part, namely $F_p = \mu A_p + \rho N_p + \Delta t^{-1} \rho M_p$ respectively. For completeness we recall that the pressure space mass matrix is given by

$$M_{p;i,j} = \int_\Omega \psi_j\,\psi_i. \tag{5.12}$$

This then completes the description of the (ideal) PCD inverse Schur complement approximation aside from the aforementioned issue of boundary conditions. Since we have to construct both A_p and F_p we must consider the enforcement of appropriate boundary conditions within these matrices. We note that the formulations given in (5.10) and (5.11) implicitly imposes a homogeneous Neumann condition on $\partial\Omega$ in both cases. However, we will see that this is not always a good choice.

Finally, we note that the choice of inverse Schur complement approximation (5.9) is also applicable when a stabilisation matrix $C \neq 0$ is necessary. Details are given in [71, Section 9.2.1], however, in our case this is easier to see because we have maintained the use of the continuous form of the operators, thus we need only be concerned with the stabilisation upon discretisation at the final stage of manipulation. For ease, we consider discretising the continuous Schur complement operator as opposed to its inverse. When we discretise (5.2) using an *inf–sup* stable discretisation we obtain

$$M_p^{-1} B F^{-1} B^T \approx F_p^{-1} A_p, \tag{5.13}$$

which gives an approximation to the Schur complement $S = BF^{-1}B^T$. However, upon using a stabilised discretisation the expression becomes

$$M_p^{-1} \left(BF^{-1}B^T + C\right) \approx F_p^{-1} A_p, \tag{5.14}$$

so long as we use a stable discretisation of $\mathcal{B}\mathcal{B}^T$, such as that given in (5.10). Thus we see that (5.9) remains appropriate in the stabilised case when $S = BF^{-1}B^T + C$.

Before turning to the consideration of boundary conditions we will first discuss some aspects of the mappings between spaces that the operators provide. We consider the steady problem here for clarity. Our goal with the PCD approach is to consider approximately commuting operators in order to yield an approximation $\widehat{S}$ to the Schur complement S. We recall that the infinite dimensional Schur complement (Poincaré–Steklov) operator is an L^2 map, that is we have $\mathcal{S} : L^2(\Omega) \to L^2(\Omega)$ given by

$$\mathcal{S} = \nabla \cdot (-\mu (\nabla \cdot \nabla) + \rho \boldsymbol{w} \cdot \nabla)_{\boldsymbol{u}}^{-1} \nabla, \tag{5.15}$$

where $(-\mu (\nabla \cdot \nabla) + \rho \boldsymbol{w} \cdot \nabla)_{\boldsymbol{u}}^{-1}$ is the solution operator for the vector (thus the subscript $\boldsymbol{u}$ notation) convection–diffusion problem with the corresponding boundary conditions of the Navier–Stokes problem. To be clearer, following [165], for $p \in L^2(\Omega)$ let $\boldsymbol{v} = (-\mu (\nabla \cdot \nabla) + \rho \boldsymbol{w} \cdot \nabla)_{\boldsymbol{u}}^{-1} \nabla p \in H^1(\Omega)$, then we can reformulate the relation $q = \mathcal{S}p$ using the intermediary problem

$$-\mu (\nabla \cdot \nabla \boldsymbol{v}) + \rho \boldsymbol{w} \cdot \nabla \boldsymbol{v} - \nabla p = \boldsymbol{0} \qquad \text{in } \Omega, \tag{5.16a}$$

$$\boldsymbol{v} = \boldsymbol{0} \qquad \text{on } \partial\Omega_a, \tag{5.16b}$$

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{0} \qquad \text{on } \partial\Omega_b, \tag{5.16c}$$

$$\boldsymbol{v} \cdot \boldsymbol{n} = 0 \text{ and } \boldsymbol{t} \cdot \boldsymbol{\sigma} \cdot \boldsymbol{n} = 0 \qquad \text{on } \partial\Omega_c, \tag{5.16d}$$

$$\boldsymbol{v} \cdot \boldsymbol{t} = 0 \text{ and } \boldsymbol{n} \cdot \boldsymbol{\sigma} \cdot \boldsymbol{n} = 0 \qquad \text{on } \partial\Omega_d, \tag{5.16e}$$

with q then being given from $\boldsymbol{v}$ as $q = \nabla \cdot \boldsymbol{v}$. On the other hand, the PCD approximate Schur complement operator is given in a similar way by

$$\widehat{\mathcal{S}} = (-\mu (\nabla \cdot \nabla) + \rho \boldsymbol{w} \cdot \nabla)_{p}^{-1} \nabla \cdot \nabla, \tag{5.17}$$

where the subscript p emphasises this requires the solution to a scalar problem. In order to be a well defined map in a weak sense we require an input argument from $H^1(\Omega)$, a more regular space than $L^2(\Omega)$. To be concrete, this involves operators $\mathcal{A}_p : H^1(\Omega) \to L^2(\Omega)$ and $\mathcal{F}_p : H^1(\Omega) \to L^2(\Omega)$. Note further that this requirement provides the ability to take the trace operator and so consider boundary conditions on $\partial\Omega$ (the trace belonging to the Sobolev space $H^{\frac{1}{2}}(\partial\Omega)$; see [80]). We remark that in practice these issues can be circumvented when using the discontinuous pressure approximation we consider ($\boldsymbol{P}_{-1}$ elements) by treating functions elementwise; see Section 5.4.2.

Turning to the commutator $\mathcal{E}$ in (5.1), then formally the argument is required to be regular enough so that we can take appropriate third derivatives, which need not be true for the functions in the velocity space $\boldsymbol{H}^1(\Omega)$. Thus we see that the notion of commutation used is implicitly requiring higher regularity of both the velocity and pressure. Nevertheless, this heuristic approach is effective.

5.1.2 Boundary conditions within the PCD preconditioner

The issue of appropriate boundary conditions for F_p and A_p is an important one. Some early work is found in [165]. The approach we follow was first proposed in [72] for F_p and then subsequently developed in [71] for both A_p and F_p. We start by considering the pressure space convection–diffusion operator in the stationary case to introduce the main concepts.

It is constructive to first consider the commutator in 1D. In this case we have

$$\mathcal{B}\mathcal{F}u = \frac{\mathrm{d}}{\mathrm{d}x}\left(-\mu\,\frac{\mathrm{d}^2u}{\mathrm{d}x^2} + \rho\,w\frac{\mathrm{d}u}{\mathrm{d}x}\right) = -\mu\,\frac{\mathrm{d}^3u}{\mathrm{d}x^3} + \rho\,w\frac{\mathrm{d}^2u}{\mathrm{d}x^2} = \mathcal{F}_p\mathcal{B}u, \tag{5.18}$$

where $\mathcal{F}_p = \mathcal{F}$ (except perhaps in its boundary conditions). Ideally we would like the commutation in (5.18), assuming w is constant, to hold not just in the interior of Ω but on the boundary $\partial\Omega$ too. Consider a problem with inflow and outflow boundary conditions. We suppose that $\mathcal{F}$ is defined on $\Omega = (0,1)$ and comes equipped with the boundary conditions $u(0) = 0$ (inflow) and $\mu u'(1) = 0$ (an outflow condition). In addition we assume, from the left-to-right nature of the flow, the boundary condition $u(0) = 0$ for $\mathcal{B}$. Thus the composite operator $\mathcal{B}\mathcal{F}$ will have the boundary conditions

$$u(0) = 0, \quad \mu u'(1) = 0, \quad \text{and} \quad -\mu u''(0) + \rho w u'(0) = 0. \tag{5.19}$$

The first two come from $\mathcal{F}$ while the final condition ensures that $\mathcal{F}u$ is an appropriate argument for $\mathcal{B}$. For the commutation in (5.18) to hold on the boundary as well as the interior of Ω we need $\mathcal{F}_p$ to have the boundary conditions

$$\mu u(1) = 0 \quad \text{and} \quad -\mu u'(0) + \rho w u(0) = 0. \tag{5.20}$$

This can be seen by taking one derivative (now associated with $\mathcal{B}$) off the final two conditions in (5.19) so as to ensure that $\mathcal{B}\mathcal{F} = \mathcal{F}_p\mathcal{B}$ on the boundary.

In higher dimensions the appropriate commutation is given in (5.5) with the (scalar) pressure space operator being

$$\mathcal{F}_p = -\mu\,(\nabla\cdot\nabla) + \rho\,\boldsymbol{w}\cdot\nabla\,. \tag{5.21}$$

However, as the analysis in [72] shows, the commutator in 2D (and thus in higher dimensions) cannot be zero everywhere on the boundary for inflow–outflow problems. Instead, it is seen that appropriate boundary conditions for $\mathcal{F}_p$ acting on a scalar function p are those corresponding to the 1D case, namely a Dirichlet condition at the outflow $\partial\Omega_{\text{out}}$ and a Robin condition at the inflow $\partial\Omega_{\text{in}}$ given by

$$\mu p = 0 \text{ on } \partial\Omega_{\text{out}} \quad \text{and} \quad -\mu\,\frac{\partial p}{\partial n} + \rho\,\boldsymbol{w}\cdot\boldsymbol{n}\,p = 0 \text{ on } \partial\Omega_{\text{in}}. \tag{5.22}$$

Note that this promotes the normal components of the commutator at the boundary, ensuring they are zero. There are too many conditions to ensure all components are zero but the choice of boundary conditions in (5.22) is seen to be the most favourable choice. We remark that the choice to promote the normal component makes sense in n-D for $n \geq 3$ since we can only promote one component and there are $n-1$ tangent directions and only one normal direction. For the more general boundary conditions (2.11) that we consider, we advocate the choice corresponding to the normal boundary condition, that is a Dirichlet condition on the outflow boundary $\partial\Omega_b \cup \partial\Omega_d$ for F_p and a Robin condition on the inflow or wall boundary $\partial\Omega_a \cup \partial\Omega_c$.

Remark 6 (Implementing the Robin condition) *A clever way to implement the Robin condition within a finite element framework is to apply integration by parts to the operator in its weak form to give*

$$\mathcal{F}_p(p,q) = \int_\Omega \mu \nabla p \cdot \nabla q - \int_\Omega \rho\, (\boldsymbol{w} \cdot \nabla q)\, p + \int_{\partial\Omega} \left(-\mu \frac{\partial p}{\partial n} + \rho\, \boldsymbol{w} \cdot \boldsymbol{n}\, p \right) q, \quad (5.23)$$

as $\boldsymbol{w}$ is divergence free. Note that strictly speaking $\boldsymbol{w}$ will only be discretely divergence free in general since it is the previous approximation to the velocity in the Picard iteration, as such this is only a heuristic argument. However, since we already made the stronger assumption that $\boldsymbol{w}$ was constant in the derivation of $\mathcal{F}_p$, this is not of any particular concern. Now, by setting the boundary term in (5.23) *to be zero, we enforce the appropriate Robin condition in* (5.22)*. This novel strategy provides a simple and effective way to impose the Robin condition on the inflow boundary, a condition which has previously been more complicated to enforce and required explicit treatment, as in [209]. Note that we must still explicitly enforce the Dirichlet condition from* (5.22)*.*

The operator A_p does not come from aiming to satisfy a commutator relationship but instead as the discretisation of $\mathcal{B}\mathcal{B}^T$. To derive boundary conditions on A_p we go back to considering the divergence operator $\mathcal{B}$ and for simplicity consider the 1D case. We suppose again that $\mathcal{B}$ comes equipped with a homogeneous Dirichlet condition on the inflow boundary. Now the *adjoint* operator $\mathcal{B}^T$ takes on *adjoint* boundary conditions, namely a homogeneous Dirichlet condition on the remaining outflow boundary. Hence the operator $\mathcal{B}\mathcal{B}^T$ must have a Neumann condition on the inflow boundary and a Dirichlet condition on the outflow boundary, that is

$$p = 0 \text{ on } \partial\Omega_{\text{out}} \quad \text{and} \quad \frac{\partial p}{\partial n} = 0 \text{ on } \partial\Omega_{\text{in}}. \quad (5.24)$$

As such, we prescribe for A_p a Dirichlet condition on the outflow boundary $\partial\Omega_b \cup \partial\Omega_d$ and a Neumann condition on the inflow or wall boundary $\partial\Omega_a \cup \partial\Omega_c$. The same conclusions, though derived in a somewhat different manner, are found in [71].

We conclude that the boundary conditions which must be explicitly enforced, heeding Remark 6, are the Dirichlet conditions on F_p and A_p on the outflow boundary $\partial\Omega_b \cup \partial\Omega_d$. Note that for enclosed flow there are no outflow boundaries and so no boundary modifications are required from the standard formulation of PCD given in (5.9)–(5.12). Further, the Robin condition reduces to the (natural) Neumann condition since $\boldsymbol{w}\cdot\boldsymbol{n} = 0$. Finally, we note that in the time-dependent case the primary (spatial) differential operators involved remain the same and the time-stepping term should be added in after consideration of the boundary conditions. As such, the same conditions (5.22)–(5.24) are appropriate and no further modifications are required.

Before moving on, we note that we will also consider the 'single-phase PCD' approach for the *two-phase* problem. By this we mean using (5.9) with F_p as in (5.11) except that ρ and μ must now be moved inside the integrals (see (5.28)).

5.1.3 A relation with the Cahouet–Chabard preconditioner

We conclude this section by noting a relation between the PCD preconditioner (5.9) and the Cahouet–Chabard preconditioner [42] for the generalised Stokes problem. We note this relation has been observed several times in the literature, seemingly independently, for instance in [140, 215]. This comes about by our observation that the Cahouet–Chabard preconditioner can also be derived by a commutator argument; as discussed in Section 4.5. The fact that PCD generalises the Cahouet–Chabard preconditioner to the Navier–Stokes case will provide a useful viewpoint on how a two-phase PCD preconditioner should behave.

In Stokes flow the convective term is omitted so that in (3.37) for single-phase flow $F = \mu A + \Delta t^{-1}\rho M$, where A is the discrete velocity Laplacian and M is the velocity mass matrix. Cahouet and Chabard show that the appropriate Schur complement which balances these two terms is

$$\widehat{S}^{-1} = \mu M_p^{-1} + \Delta t^{-1}\rho A_p^{-1}, \tag{5.25}$$

where A_p is a discrete pressure Laplacian operator, for instance given by (5.10) for continuous pressure approximation or $BM^{-1}B^T$, as in (4.16). Note that in (4.16) the formulation is for that in (2.23) which uses the kinematic viscosity ν; to keep the comparison with PCD clear we have kept with the dynamic viscosity μ here and thus the inverse Schur complement is correspondingly multiplied by the density ρ.

Now consider the PCD preconditioner in the time-dependent Stokes case. Since the convective term is not present we have $F_p = \mu A_p + \Delta t^{-1} \rho M_p$ and thus the inverse Schur complement approximation

$$\widehat{S}^{-1} = A_p^{-1} F_p M_p^{-1} = A_p^{-1} \left(\mu A_p + \Delta t^{-1} \rho M_p \right) M_p^{-1} = \mu M_p^{-1} + \Delta t^{-1} \rho A_p^{-1}. \qquad (5.26)$$

Hence we see that the PCD preconditioner precisely reduces to the Cahouet–Chabard preconditioner in the case of generalised Stokes flow. This provides the viewpoint that PCD extends the Cahouet–Chabard preconditioner to the case of Navier–Stokes flow.

5.2 Two-phase PCD through two-phase Cahouet–Chabard

On its first introduction, the PCD preconditioner for steady flow was derived using Fourier techniques and Green's tensors [118]. Due to the variable coefficient nature of two-phase flow, this approach cannot apply here. However, in the original approach, in order that the preconditioner defaults to the optimal choice in the Stokes limit when the convective term tends to zero, a mass matrix is included to give the correct scaling. The same philosophy applies here.

Before addressing the appropriate choice in the Stokes case, we must be clear on the form of the viscous term used. In the literature different forms are considered, including $-\nabla \cdot (\mu \nabla \boldsymbol{u})$ in [164], $-\nabla \cdot (\mu \boldsymbol{D}\boldsymbol{u})$ in [90, 161], and $-\nabla \cdot (2\mu \boldsymbol{D}\boldsymbol{u})$ in [91, 43]. The overall scaling of the latter term contains an additional factor of two compared with the former choices, as such an appropriate Schur complement approximation for the Stokes problem differs by a factor of a half. The distinction between different forms of the viscous term used (including incorporating bulk viscosity) and the effect on the Schur complement is discussed further in [43]. Since we use the latter choice above, we present results for this case, including a factor of two where appropriate as compared with the cited references.

For two-phase flow, the appropriate pressure mass matrix arising in the Stokes problem (2.19) is given in [164] and scales inversely with the viscosity as

$$M_{p;i,j}^{(1/\mu)} = \int_\Omega (2\mu)^{-1} \psi_j \psi_i. \qquad (5.27)$$

This mass matrix is also suitable for general variable viscosity Stokes flows, as seen in the analysis of [90].

Looking at the construction of the PCD preconditioner in Section 5.1.1, the natural generalisation of F_p in (5.11) for piecewise constant density and viscosity is given by

$$F_{p;i,j} = \int_\Omega \mu \nabla \psi_j \cdot \nabla \psi_i + \int_\Omega \rho \left(\boldsymbol{w} \cdot \nabla \psi_j \right) \psi_i + \Delta t^{-1} \int_\Omega \rho \psi_j \psi_i. \tag{5.28}$$

If we assume that PCD still takes the form in (5.9) with (5.28) but additional scaling in the other terms, then, so that the viscosity scaling within the Schur complement is commensurate when we use the scaled mass matrix $M_p^{(1/\mu)}$, we require the scaled pressure Laplacian

$$A_{p;i,j}^{(\mu)} = \int_\Omega \mu \nabla \psi_j \cdot \nabla \psi_i. \tag{5.29}$$

Note this is already constructed as part of F_p just as in the single-phase case since for constant viscosity $A_p^{(\mu)} = \mu A_p$. These choices give a Schur complement approximation

$$\widehat{S}^{-1} = \left(A_p^{(\mu)} \right)^{-1} F_p \left(M_p^{(1/\mu)} \right)^{-1}. \tag{5.30}$$

However, from our numerical experience, it is apparent that the performance of this preconditioner can depend poorly on the density ratio of the two fluids. In particular, (5.30) does not work so well in the time-dependent case. To understand this we go back to the relationship with the Cahouet–Chabard preconditioner detailed in Section 5.1.3.

By considering an abstract parameter dependent saddle point system, Olshanskii, Peters and Reusken [161] derive a preconditioner for two-phase time-dependent Stokes flow which can be seen as generalising the Cahouet–Chabard preconditioner. For our scaling of the Stokes equations in (2.19) (utilising the dynamic viscosity as opposed to the kinematic viscosity), this preconditioner uses an approximation to the inverse Schur complement given by

$$\widehat{S}^{-1} = \left(M_p^{(1/\mu)} \right)^{-1} + \Delta t^{-1} \left(A_p^{(1/\rho)} \right)^{-1}, \tag{5.31}$$

where the pressure Laplacian is now inversely proportional to the density

$$A_{p;i,j}^{(1/\rho)} = \int_\Omega \rho^{-1} \nabla \psi_j \cdot \nabla \psi_i. \tag{5.32}$$

The new feature here is the dependence on the density ρ within the Laplacian term. This "two-phase Cahouet–Chabard" approximation can be understood heuristically by considering the single-phase approach, appropriately scaled in our case as in (5.25), and incorporating the variable coefficients (μ and ρ) into the integral definitions of the matrices to obtain (5.31).

We note that an inverse Schur complement approximation very similar to (5.31), appropriate for finite volume Stokes solvers, is given in [43]. Here the MAC (marker-and-cell) discretisation on staggered grids is employed. The primary difference is that for finite volumes a diagonal matrix of viscosities at each pressure degree of freedom can be used for the viscous term, whereas in the finite element setting a mass matrix is appropriate.

It is now clear that, from the viewpoint of the relation between PCD and the Cahouet–Chabard preconditioner, the approximation (5.30) does not default to the correct choice of the preconditioner (5.31) in the time-dependent Stokes case. Instead, (5.30) reduces to the incorrect approximation

$$\widehat{S}^{-1} = {M_p^{(1/\mu)}}^{-1} + \Delta t^{-1} \left(A_p^{(\mu)}\right)^{-1} M_p^{(\rho)} \left(M_p^{(1/\mu)}\right)^{-1}. \tag{5.33}$$

To give a more robust generalisation of PCD to two-phase flow with appropriate scaling we must split the matrix F_p and treat the separate terms accordingly.

The matrix F_p in the two-phase case consists of the three parts

$$F_p = A_p^{(\mu)} + N_p^{(\rho)} + \Delta t^{-1} M_p^{(\rho)}. \tag{5.34}$$

To treat the viscous part $A_p^{(\mu)}$ of F_p we use the scaling from (5.27) and (5.29) but to treat the remaining part, depending on the density, we scale using the mass matrix

$$M_{p;i,j}^{(\rho)} = \int_\Omega \rho\, \psi_j\, \psi_i \tag{5.35}$$

and use the scaled pressure Laplacian of (5.32). This yields the approximation

$$\widehat{S}^{-1} = \left(M_p^{(1/\mu)}\right)^{-1} + \left(A_p^{(1/\rho)}\right)^{-1} \left(N_p^{(\rho)} + \Delta t^{-1} M_p^{(\rho)}\right) \left(M_p^{(\rho)}\right)^{-1}. \tag{5.36}$$

In practice, however, we have found it beneficial to further cancel the scaling with the density ρ in the final two bracketed terms of (5.36) to give two-phase PCD as

$$\widehat{S}^{-1} = \left(M_p^{(1/\mu)}\right)^{-1} + \left(A_p^{(1/\rho)}\right)^{-1} \left(N_p^{(1)} + \Delta t^{-1} M_p^{(1)}\right) \left(M_p^{(1)}\right)^{-1}. \tag{5.37}$$

Here the bracketed exponent is given as 1 to be clear that these are unscaled terms whose definition does not include ρ. Namely we have

$$N_{p;i,j}^{(1)} = \int_\Omega (\boldsymbol{w} \cdot \nabla\, \psi_j)\, \psi_i, \tag{5.38}$$

with $M_p^{(1)}$ being the usual mass matrix. While this gives a different approximation, the overall scaling of the two separate terms in the sum remains the same and, although

it is feasible to cancel the density as in (5.37), it remains important that $A_p^{(1/\rho)}$ keeps the correct scaling. Further, note that $M_p^{(1)}$ remains the same at each time-step while $M_p^{(\rho)}$ would change due to the moving phases and so would need recomputing. From the arguments presented here, it is not clear that (5.37) should be preferable to (5.36). However, we shall see that the choice of scaling can be derived by considering appropriately weighted commutators. We also remark that when using the viscous term $-\nabla \cdot (2\mu \boldsymbol{D})$ the matrix $M_p^{(1/\mu)}$ features the factor of a half as in (5.27).

While utilising the relation with Cahouet–Chabard is not fully prescriptive for a two-phase PCD preconditioner, it makes clear that a straightforward adaptation of the single-phase PCD preconditioner is incorrect. We emphasise that the generalisation of PCD to two-phase flow in (5.37) (or (5.36)) cannot be written in the form of the original PCD in (5.9) with scaling in A_p and M_p and the natural generalisation of F_p in (5.28). As such, the form (5.9), where F_p is the pressure space analogue of F, is a simplification which can only be made for an everywhere constant density and viscosity flow.

As with single-phase PCD, in the two-phase PCD approximation of (5.37) only a single pressure Laplacian solve is needed, this is because the appearance of $A_p^{(\mu)}$ cancels out. We note that in the time-dependent Stokes case, when $N_p^{(1)}$ is zero, we have

$$\begin{aligned} \widehat{S}^{-1} &= \left(M_p^{(1/\mu)}\right)^{-1} + \left(A_p^{(1/\rho)}\right)^{-1} \left(\Delta t^{-1} M_p^{(1)}\right) \left(M_p^{(1)}\right)^{-1} \\ &= \left(M_p^{(1/\mu)}\right)^{-1} + \Delta t^{-1} \left(A_p^{(1/\rho)}\right)^{-1}, \end{aligned} \tag{5.39}$$

and so we return to the generalised Cahouet–Chabard preconditioner (5.31). Further, despite the new form of the preconditioner (5.37), it is a generalisation of the original single-phase PCD preconditioner (5.9) since for everywhere constant viscosity μ and density ρ (and thus using the Laplacian for the viscous term, removing the factor of a half in (5.27))

$$\begin{aligned} \widehat{S}^{-1} &= \left(\mu^{-1} M_p\right)^{-1} + (\rho^{-1} A_p)^{-1} \left(N_p + \Delta t^{-1} M_p\right) M_p^{-1} \\ &= A_p^{-1} \left(\mu A_p\right) M_p^{-1} + A_p^{-1} \left(\rho N_p + \Delta t^{-1} \rho M_p\right) M_p^{-1} \\ &= A_p^{-1} F_p M_p^{-1}. \end{aligned} \tag{5.40}$$

While we have considered discretisation with finite elements, we envisage that the two-phase PCD approach can also be utilised with finite volumes or finite differences. For instance, using the MAC discretisation [99] and building on the work in [43]. The primary requirement is an appropriate discretisation of the pressure space convection term.

5.3 Weighted commutator derivation of two-phase PCD and appropriate boundary conditions

5.3.1 Introducing the need for weighted commutators

We now consider commutators for the variable coefficient equations (2.16) of interest. For two-phase flow the density and viscosity are functions of space, namely piecewise constant if the interface is not smoothed. We firstly note that, mathematically, nothing in our arguments would change if we were to have multi-phase flow or fully spatially varying density and viscosity. We first consider steady flow, in which case the convection–diffusion operator is

$$\mathcal{F} = -\nabla \cdot (2\mu\, \boldsymbol{D}) + \rho\, \boldsymbol{w} \cdot \nabla\,, \tag{5.41}$$

where we recall that $\boldsymbol{D}$ is the rate of deformation tensor, $\boldsymbol{D}\boldsymbol{u} = \frac{1}{2}\left(\nabla\, \boldsymbol{u} + (\nabla\, \boldsymbol{u})^T\right)$. We again look at the one-dimensional case first to gain an understanding of the scaling necessary to aid commutation and for ease of exposition consider the simplified form of the viscous term $-\nabla \cdot \mu\nabla$.

In 1D the commutator (5.1) now becomes

$$\mathcal{B}\mathcal{F}u = \frac{\mathrm{d}}{\mathrm{d}x}\left(-\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}u}{\mathrm{d}x} + \rho\, w\frac{\mathrm{d}u}{\mathrm{d}x}\right) = -\frac{\mathrm{d}^2}{\mathrm{d}x^2}\mu\frac{\mathrm{d}u}{\mathrm{d}x} + \frac{\mathrm{d}}{\mathrm{d}x}\rho\, w\frac{\mathrm{d}u}{\mathrm{d}x} = \mathcal{F}_p\mathcal{B}u. \tag{5.42}$$

This suggests, assuming w constant as before, that $\mathcal{F}_p$ should be

$$\mathcal{F}_p = -\frac{\mathrm{d}^2}{\mathrm{d}x^2}\mu + w\frac{\mathrm{d}}{\mathrm{d}x}\rho. \tag{5.43}$$

With μ and ρ no longer constant we cannot pass them through the derivatives and, neglecting boundary terms, the weak form of $\mathcal{F}_p$ would read

$$\mathcal{F}_p(p,q) = \int_\Omega \frac{\mathrm{d}}{\mathrm{d}x}(\mu p)\frac{\mathrm{d}q}{\mathrm{d}x} + \int_\Omega w\frac{\mathrm{d}}{\mathrm{d}x}(\rho p)\, q. \tag{5.44}$$

The problem here is that we are seemingly required to take derivatives of the viscosity and density, which may be piecewise constant. Further, if we remove this obstacle by passing μ and ρ through the derivatives to use the same formulation as in the single-phase case, namely yielding (5.9) with F_p given by (5.28) (in the time-dependent case), then the preconditioner significantly deteriorates in performance as the viscosity and density contrasts increase (as we shall see in later results). As such, we see that the arguments used to derive the original single-phase PCD approach fail due to the variable coefficients.

However, the use of commutators is not a complete loss. Instead, we must consider a modification to the commutator by including an additional scaling through weight functions designed to assist in the commutation. In particular, consider

$$\mathcal{B}\mu\mathcal{F}u = \frac{\mathrm{d}}{\mathrm{d}x}\mu\left(-\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}u}{\mathrm{d}x} + \rho w\frac{\mathrm{d}u}{\mathrm{d}x}\right) = -\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}u}{\mathrm{d}x} + \frac{\mathrm{d}}{\mathrm{d}x}\mu\rho w\frac{\mathrm{d}u}{\mathrm{d}x} = \mathcal{F}_p\mu\mathcal{B}u, \quad (5.45)$$

where now

$$\mathcal{F}_p = -\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}}{\mathrm{d}x} + \frac{\mathrm{d}}{\mathrm{d}x}\rho w. \quad (5.46)$$

The viscous part of $\mathcal{F}_p$ is now the same as for $\mathcal{F}$ but the convective part is not. Note that the use of the viscosity scaling in the commutator does not change the resulting boundary conditions for $\mathcal{F}_p$; they are still those in (5.22). To deal with the convective part of $\mathcal{F}_p$ we consider the weak formulation where integration by parts is used for both terms; see Remark 6. Now, setting the Robin boundary term to zero, this gives

$$\mathcal{F}_p(p,q) = \int_\Omega \mu\,\frac{\mathrm{d}p}{\mathrm{d}x}\,\frac{\mathrm{d}q}{\mathrm{d}x} - \int_\Omega \rho\, w\,\frac{\mathrm{d}q}{\mathrm{d}x}\,p. \quad (5.47)$$

We note that, as well as naturally enforcing the appropriate Robin condition on the inflow boundary, this use of integration by parts has allowed us to remove all derivatives from μ and ρ to gain a legitimate weak form for $\mathcal{F}_p$. We further remark that in this one-dimensional setting we did not use the assumption that w was constant in the derivation, thus we have a perfect vanishing commutator in the continuum case.

We will discuss what happens in higher dimensions soon but for now we assume the natural generalisation of (5.47) and that the viscous operator in $\mathcal{F}$ is $-\nabla\cdot\mu\nabla$. We follow the same procedure as for single-phase PCD to obtain an inverse Schur complement approximation. We start from the weighted commutator

$$\mathcal{B}\mu\mathcal{F} \approx \mathcal{F}_p\mu\mathcal{B}, \quad (5.48)$$

and formally rearrange (having applied $\mathcal{F}^{-1}\mathcal{B}^T$ on the right first) to give

$$\left(\mathcal{B}\mathcal{F}^{-1}\mathcal{B}^T\right)^{-1} \approx (\mathcal{B}\mu\mathcal{B}^T)^{-1}\mathcal{F}_p\mu. \quad (5.49)$$

Discretisation in the same manner as (5.8) allows us to obtain the approximation

$$\widehat{S}^{-1} = \left(A_p^{(\mu)}\right)^{-1} F_p \left(M^{(1/\mu)}\right)^{-1}. \quad (5.50)$$

We recognise this as "viscosity scaled" PCD (5.30) devised in order to balance the viscosity scaling from the appropriate mass matrix. Now, our knowledge on the appropriate boundary conditions gives

$$F_{p;i,j} = \int_\Omega \mu\,\nabla\,\psi_j\cdot\nabla\,\psi_i - \int_\Omega \rho\,(\boldsymbol{w}\cdot\nabla\,\psi_i)\,\psi_j. \quad (5.51)$$

However, when incorporating the time-dependent term we saw previously in (5.33) that this approach results in an incorrect approximation of the time-stepping term. This issue also becomes apparent in the weighted commutator (5.45) when trying to incorporate the time-stepping term since

$$\mathcal{B}\mu\mathcal{T}\,u = \frac{\mathrm{d}}{\mathrm{d}x}\mu\,\Delta t^{-1}\rho\,u \neq \Delta t^{-1}\rho\,\mu\,\frac{\mathrm{d}u}{\mathrm{d}x} = \mathcal{T}_p\mu\mathcal{B}u. \tag{5.52}$$

We will ultimately derive an approach which can rectify this issue and correctly reduce to the generalised Cahouet–Chabard preconditioner (5.31). However, in order to gain more understanding on appropriate commutators first, we now consider the individual operators which comprise $\mathcal{F}$ and which we seek to commute with the divergence. Namely we have the components $\mathcal{F} = \mathcal{A}^{(\mu)} + \mathcal{N}^{(\rho)} + \mathcal{T}^{(\rho)}$ corresponding the diffusive, convective, and time-stepping terms respectively. We examine each term separately, akin to the approach in [228, Section 2.3], and generalise from 1D to higher dimensions.

5.3.2 Split weighted commutators

We first observe that the two-phase time-stepping term $\mathcal{T}^{(\rho)}$ will only commute with the choice of an inverse density weighting in the commutator to give

$$\mathcal{B}\rho^{-1}\mathcal{T}^{(\rho)}\boldsymbol{u} = \nabla\cdot\left(\rho^{-1}\Delta t^{-1}\rho\boldsymbol{u}\right) = \Delta t^{-1}\nabla\cdot\boldsymbol{u} = \mathcal{T}_p^{(\rho)}\rho^{-1}\mathcal{B}\boldsymbol{u}, \tag{5.53}$$

with $\mathcal{T}_p^{(\rho)}$ being the same scaled identity operator as $\mathcal{T}^{(\rho)}$ but acting on the pressure space. The choice of this density weighting allows us to remove the dependence on density in the commutator. Further, while in the 1D case the convective term $\mathcal{N}^{(\rho)}$ commutes using any weight (as can be observed from (5.45)), in higher dimensions to have the commutator exactly vanish (assuming $\boldsymbol{w}$ constant) in the interior of Ω we must also select an inverse density weighting and again remove the dependence on density in the commutator. Thus we obtain

$$\mathcal{B}\rho^{-1}\mathcal{N}^{(\rho)}\boldsymbol{u} = \nabla\cdot\left((\boldsymbol{w}\cdot\nabla\,)\,\boldsymbol{u}\right) = (\boldsymbol{w}\cdot\nabla\,)\,\nabla\cdot\boldsymbol{u} = \mathcal{N}_p^{(\rho)}\rho^{-1}\mathcal{B}\boldsymbol{u}, \tag{5.54}$$

where

$$\mathcal{N}_p^{(\rho)} = (\boldsymbol{w}\cdot\nabla\,)\,\rho. \tag{5.55}$$

Finally, for the diffusive term $\mathcal{A}^{(\mu)}$ in higher dimensions we can no longer have the commutator vanish in the interior of Ω, despite the choice of a viscosity weighting to aid the commutation. This is because the derivatives in different directions can no

longer commute due to the scalar functions of μ present. Furthermore, there is no analogue of the rate of deformation tensor $\boldsymbol{D}$ on the pressure space. Nonetheless even if we suppose for simplification that we consider the scaled Laplacian $-(\nabla \cdot \mu \nabla)$ for the diffusive part, which can act on scalars as well as componentwise on vectors, we still do not commute exactly as

$$\mathcal{B}\mu\mathcal{A}^{(\mu)}\boldsymbol{u} = \nabla \cdot \mu \left(-\nabla \cdot \mu \nabla\right) \boldsymbol{u} \neq -\left(\nabla \cdot \mu \nabla\right) \mu \nabla \cdot \boldsymbol{u} = \mathcal{A}_p^{(\mu)} \mu \mathcal{B}\boldsymbol{u}, \tag{5.56}$$

with $\mathcal{A}_p^{(\mu)}$ given by the pressure space analogue $-(\nabla \cdot \mu \nabla)$. Despite this, what we have is that the number of derivatives on the μ factors in the approximate commutator are equal, even if the derivatives act in different directions.

Before we address construction of an inverse Schur complement approximation, we discuss the issue of appropriate boundary conditions within the pressure space operators introduced above in light of our consideration of weighted commutators. Analogous arguments to those providing conditions on $\mathcal{F}_p$ in the 1D single-phase case apply to $\mathcal{A}_p^{(\mu)}$ by dropping the convective term in (5.22). As such we take it have a Neumann condition on the inflow boundary and a Dirichlet condition on the outflow boundary. We note these are precisely the boundary conditions on $\mathcal{B}\mathcal{B}^T$ in (5.24).

For boundary conditions on $\mathcal{N}_p^{(\rho)}$ it is not quite so straightforward to apply the same arguments as for $\mathcal{F}_p$ before since now, dropping the diffusive term, we only have a first order differential operator. Nonetheless, if we assume that we can take a limit where μ goes to zero in (5.22) (we could see this as a regularisation of the convective term where μ is a small parameter adding in some diffusion) then we see that the inflow and outflow conditions effectively match to yield a homogeneous Dirichlet condition on the boundary. Alternatively we can consider again the 1D case, with $\Omega = (0, 1)$, and the commutator

$$\mathcal{B}\rho^{-1}\mathcal{N}^{(\rho)}\boldsymbol{u} = \frac{\mathrm{d}}{\mathrm{d}x} w \frac{\mathrm{d}u}{\mathrm{d}x} = \mathcal{N}_p^{(\rho)} \rho^{-1} \mathcal{B}\boldsymbol{u}. \tag{5.57}$$

If we take the inflow conditions $u(0) = 0$ for both $\mathcal{B}$ and $\mathcal{N}^{(\rho)}$ then the composite operator $\mathcal{B}\rho^{-1}\mathcal{N}^{(\rho)}$ has boundary conditions

$$u(0) = 0 \quad \text{and} \quad wu'(0) = 0. \tag{5.58}$$

Hence, so that $\mathcal{N}_p^{(\rho)}\rho^{-1}\mathcal{B}$ also takes on these boundary conditions we must apply a homogeneous Dirichlet condition $u(0) = 0$ for $\mathcal{N}_p^{(\rho)}$. Further suggestion that a Dirichlet condition is appropriate stems from the idea that integration by parts yields

the correct *natural* condition in the weak form of the operator (as was true in the single-phase case). The weak formulation of $\mathcal{N}_p^{(\rho)}$ can be written as

$$\begin{aligned}\mathcal{N}_p^{(\rho)}(p,q) &= \int_\Omega q\,(\boldsymbol{w}\cdot\nabla)\,\rho p = -\int_\Omega \rho p \nabla\cdot(q\boldsymbol{w}) + \int_{\partial\Omega}\rho\,(\boldsymbol{w}\cdot\boldsymbol{n})\,pq \\ &= -\int_\Omega \rho p\,(\boldsymbol{w}\cdot\nabla)\,q + \int_{\partial\Omega}\rho\,(\boldsymbol{w}\cdot\boldsymbol{n})\,pq,\end{aligned} \tag{5.59}$$

since $\nabla\cdot(q\boldsymbol{w}) = (\boldsymbol{w}\cdot\nabla)\,q$ due to the fact that $\boldsymbol{w}$ is divergence free (or at least approximately; see Remark 6). Setting the boundary term in (5.59) to be zero again enforces a homogeneous Dirichlet condition on $\mathcal{N}_p^{(\rho)}$. While these arguments are not fully conclusive that this condition is the correct one to apply, our numerical results will demonstrate the effectiveness of this approach. We now consider the issue of obtaining an inverse Schur complement approximation.

In order to use the above weighted commutators a technical difficulty must be faced: due to the two different choices of weighting functions, one for the diffusive term and one for the convective and time-stepping terms, we cannot piece together a direct approximation of the inverse Schur complement as in (5.3). This is made clear by collecting together the commutators as

$$\mathcal{B}\left(\mu\mathcal{A}^{(\mu)} + \rho^{-1}\mathcal{N}^{(\rho)} + \rho^{-1}\mathcal{T}^{(\rho)}\right) = \left(\mathcal{A}_p^{(\mu)}\mu + \mathcal{N}_p^{(\rho)}\rho^{-1} + \mathcal{T}_p^{(\rho)}\rho^{-1}\right)\mathcal{B}. \tag{5.60}$$

We now can no longer isolate the Schur complement due to the different weighting involved. What we can write, by keeping the two weightings separate and using the standard rearrangement procedure on each, is that

$$\left(\mathcal{B}\mathcal{A}^{(\mu)^{-1}}\mathcal{B}^T\right)^{-1} \approx (\mathcal{B}\mu\mathcal{B}^T)^{-1}\mathcal{A}_p^{(\mu)}\mu = \mu, \tag{5.61a}$$

$$\begin{aligned}\left(\mathcal{B}\left(\mathcal{N}^{(\rho)} + \mathcal{T}^{(\rho)}\right)^{-1}\mathcal{B}^T\right)^{-1} &\approx (\mathcal{B}\rho^{-1}\mathcal{B}^T)^{-1}\left(\mathcal{N}_p^{(\rho)} + \mathcal{T}_p^{(\rho)}\right)\rho^{-1} \\ &= (\mathcal{B}\rho^{-1}\mathcal{B}^T)^{-1}\left(\mathcal{N}_p^{(1)} + \mathcal{T}_p^{(1)}\right).\end{aligned} \tag{5.61b}$$

Here, in (5.61a), since the boundary conditions also match, the operators cancel and we are left with a scaling of the identity operator. Further, in (5.61b), using the definitions of $\mathcal{N}_p^{(\rho)}$ and $\mathcal{T}_p^{(\rho)}$, we see that the scaling with ρ cancels so that the convective and time-stepping terms are unscaled, as in the single-phase case; for clarity we show this with a bracketed superscript scaling of 1. Note that this does not change the boundary conditions on $\mathcal{N}_p$.

To obtain an approximation to the inverse Schur complement we now must make an ad hoc approximation. This is motivated by ensuring that, without the convective term $\mathcal{N}^{(\rho)}$, we obtain the generalised Cahouet–Chabard preconditioner (5.31), having

a form which is the sum of two inverse operators. As such, we suppose that, grouping the convective term with the time-stepping term since they utilise the same scaling,

$$\left(\mathcal{B}\mathcal{F}^{-1}\mathcal{B}^T\right)^{-1} \approx \left(\mathcal{B}\mathcal{A}^{(\mu)^{-1}}\mathcal{B}^T\right)^{-1} + \left(\mathcal{B}\left(\mathcal{N}^{(\rho)} + \mathcal{T}^{(\rho)}\right)^{-1}\mathcal{B}^T\right)^{-1}. \tag{5.62}$$

While not fully justified, we note that if $\mathcal{B}$ were invertible this approximation would be exact (neglecting technical issues such as boundary conditions). We also note the similarity with the approach given in [228, Section 2.3]. Allowing ourselves this approximation we can determine a form of two-phase PCD preconditioner.

First, discretisation of the weighted commutators in (5.61) gives

$$\left(BA^{(\mu)^{-1}}B^T\right)^{-1} \approx \left(M_p^{(1/\mu)}\right)^{-1}, \tag{5.63a}$$

$$\left(B\left(N^{(\rho)} + T^{(\rho)}\right)^{-1}B^T\right)^{-1} \approx \left(A_p^{(1/\rho)}\right)^{-1}\left(N_p^{(1)} + T_p^{(1)}\right)\left(M_p^{(1)}\right)^{-1}. \tag{5.63b}$$

Utilising a discrete form of our ad hoc approximation (5.62) then yields

$$\widehat{S}^{-1} = \left(M_p^{(1/\mu)}\right)^{-1} + \left(A_p^{(1/\rho)}\right)^{-1}\left(N_p^{(1)} + \Delta t^{-1}M_p^{(1)}\right)\left(M_p^{(1)}\right)^{-1}. \tag{5.64}$$

Here, with the boundary conditions suggested above, we have

$$N_{p;i,j}^{(1)} = -\int_\Omega \left(\boldsymbol{w}\cdot\nabla\,\psi_i\right)\psi_j, \tag{5.65}$$

the sparse pressure Laplacian $A_p^{(1/\rho)}$ is given by (5.32) and $M_p^{(1)}$ is the standard pressure mass matrix without scaling (5.12). Additionally, the Laplacian matrix $A_p^{(1/\rho)}$ must have a Dirichlet condition explicitly imposed on the outflow, identically to the single-phase case (cf. (5.24)).

We note that the form of the PCD approximation in (5.64), is the same as that which we prescribed in (5.37). However, we now have new insight into the scaling required to capture the two-phase nature of the problem which comes through the use of weighted commutators, in particular this derivation shows why the ρ dependence in the pressure space convection and time-stepping terms cancels, which was previously unclear in Section 5.2. Furthermore, this commutator approach allows us to gain a handle on appropriate boundary conditions for the pressure space operators we construct; this is key to obtaining more scalable behaviour of a preconditioner [147]. We shall see from numerical results the importance of imposing appropriate boundary conditions. However, the use of the ad hoc approximation (5.62) is somewhat dubious and thus this derivation of two-phase PCD still raises some questions. To overcome these issues we must discard our notion that $\mathcal{F}_p$ (or equally each constituent part) should be the pressure space analogue of the original operator $\mathcal{F}$. This is already hinted at within the $\mathcal{N}_p^{(\rho)}$ operator in (5.55) since we have a derivative acting on ρ which is not present in $\mathcal{N}^{(\rho)}$. Further, we must consider a bi-weighted commutator.

5.3.3 A bi-weighted commutator

In the above *split* commutators (5.53), (5.54) and (5.56) we required two different weights to allow for commutation in the 1D case and approximate commutation in higher dimensions. However, we saw that the problem with this approach is that, by splitting the commutators, it becomes difficult to derive an appropriate expression for the inverse Schur complement. To rectify this we must utilise two different weights but in a single *bi-weighted* commutator.

Consider the following bi-weighted approximate commutator

$$\mathcal{B}\pi\mathcal{F} \approx \mathcal{F}_p\eta\mathcal{B}, \tag{5.66}$$

with weights π and η. Again, it is constructive to consider the 1D case. Firstly, we note that to have the time-stepping term be able to commute, regardless of our choice of $\mathcal{F}_p$ and η we must choose $\pi = \rho^{-1}$ and so we obtain

$$\begin{aligned}\mathcal{B}\rho^{-1}\mathcal{F}u &= \frac{\mathrm{d}}{\mathrm{d}x}\rho^{-1}\left(-\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}u}{\mathrm{d}x} + \rho w\frac{\mathrm{d}u}{\mathrm{d}x} + \Delta t^{-1}\rho\, u\right)\\ &= -\frac{\mathrm{d}}{\mathrm{d}x}\rho^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}u}{\mathrm{d}x} + \frac{\mathrm{d}}{\mathrm{d}x}w\frac{\mathrm{d}u}{\mathrm{d}x} + \Delta t^{-1}\frac{\mathrm{d}u}{\mathrm{d}x}.\end{aligned} \tag{5.67}$$

Now consider removing the derivative stemming from $\mathcal{B}$ on the right. In what remains we note that viscosity μ in the diffusive term has two derivatives acting on it so if we wish to have a weak formulation of $\mathcal{F}_p$ which does not require derivatives of the density or viscosity we must choose $\eta = \mu$. This results in an *exact* commutator in the 1D case without any assumption on w, namely

$$\mathcal{B}\rho^{-1}\mathcal{F}u = -\frac{\mathrm{d}}{\mathrm{d}x}\rho^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}u}{\mathrm{d}x} + \frac{\mathrm{d}}{\mathrm{d}x}w\frac{\mathrm{d}u}{\mathrm{d}x} + \Delta t^{-1}\frac{\mathrm{d}u}{\mathrm{d}x} = \mathcal{F}_p\mu\mathcal{B}, \tag{5.68}$$

where

$$\mathcal{F}_p = -\frac{\mathrm{d}}{\mathrm{d}x}\rho^{-1}\frac{\mathrm{d}}{\mathrm{d}x} + \frac{\mathrm{d}}{\mathrm{d}x}w\mu^{-1} + \Delta t^{-1}\mu^{-1}. \tag{5.69}$$

We now see that, while $\mathcal{F}_p$ contains the same constituent parts as $\mathcal{F}$ acting on the pressure space, the scaling in these terms is quite different. We further highlight the fact that both the wind w and the variable coefficient, in this case μ^{-1}, are to the right of the derivative in the convective term. As discussed in obtaining (5.47), in constructing the weak form of $\mathcal{F}_p$ the use of integration by parts is necessary to obtain a legitimate weak form without derivatives of a discontinuous coefficient and, as we shall see, again yields the required *natural* boundary condition on the inflow

boundary. Moreover, we emphasise that this manipulation allows for the omission of the requirement that w should be constant in the one-dimensional case.

We now show that the corresponding boundary conditions for $\mathcal{F}_p$ in (5.69) fall out naturally with the same scaling as the operators involved. Again, for the discussion we remove the time-stepping term since this is not a feature present on the level of the continuous boundary conditions on the model problem. As in Section 5.1.2, we suppose $\mathcal{F}$ is defined on $\Omega = (0, 1)$ and comes equipped with the inflow–outflow boundary conditions

$$u|_{x=0} = 0 \quad \text{and} \quad \mu \left.\frac{\mathrm{d}u}{\mathrm{d}x}\right|_{x=1} = 0, \tag{5.70}$$

and equally that $\mathcal{B}$ has the boundary condition $u|_{x=0} = 0$ prescribed. Now, the composite operator $\mathcal{B}\rho^{-1}\mathcal{F}$ has the boundary conditions

$$u|_{x=0} = 0, \quad \mu \left.\frac{\mathrm{d}u}{\mathrm{d}x}\right|_{x=1} = 0, \quad \text{and} \quad \left.\left(-\rho^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mu\frac{\mathrm{d}u}{\mathrm{d}x} + w\frac{\mathrm{d}u}{\mathrm{d}x}\right)\right|_{x=0} = 0. \tag{5.71}$$

Thus $\mathcal{F}_p$ must take on the boundary conditions

$$u|_{x=1} = 0 \quad \text{and} \quad \left.\left(-\rho^{-1}\frac{\mathrm{d}u}{\mathrm{d}x} + \mu^{-1}wu\right)\right|_{x=0} = 0, \tag{5.72}$$

so that $\mathcal{B}\rho^{-1}\mathcal{F} = \mathcal{F}_p\mu\mathcal{B}$ on the boundary. Again we have a Dirichlet condition on the outflow boundary and a Robin condition on the inflow. Utilising integration by parts the weak form of the operator becomes

$$\mathcal{F}_p(p, q) = \int_\Omega \rho^{-1}\frac{\mathrm{d}p}{\mathrm{d}x}\frac{\mathrm{d}q}{\mathrm{d}x} - \int_\Omega \mu^{-1}w\frac{\mathrm{d}q}{\mathrm{d}x}p + \int_{\partial\Omega}\left(-\rho^{-1}\frac{\mathrm{d}p}{\mathrm{d}x} + \mu^{-1}wp\right)q, \tag{5.73}$$

and so setting the boundary term to be zero enables us to enforce the correct Robin boundary condition, analogous to Remark 6.

Let us now return to the general higher dimensional case and consider the viscous operator $-\nabla\cdot(2\mu\boldsymbol{D})$ in $\mathcal{F}$ as used in our model problem. As in the single-phase case we cannot commute exactly in higher dimensions but, analogously, we have the approximate commutator

$$\mathcal{B}\rho^{-1}\mathcal{F} \approx \mathcal{F}_p\mu\mathcal{B}, \tag{5.74}$$

with

$$\mathcal{F}_p = -2\,\nabla\cdot\rho^{-1}\nabla\, + \nabla\cdot\boldsymbol{w}\mu^{-1} + \Delta t^{-1}\mu^{-1}. \tag{5.75}$$

While the time-stepping term commutes exactly, the remaining terms result only in an approximate commutator, namely, using index notation, we have

$$\frac{\partial}{\partial x_i}\left(-\rho^{-1}\frac{\partial}{\partial x_j}\mu\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)+w_j\frac{\partial u_i}{\partial x_j}\right)\approx\left(-2\frac{\partial}{\partial x_j}\rho^{-1}\frac{\partial}{\partial x_j}+\frac{\partial w_j}{\partial x_j}\right)\frac{\partial u_i}{\partial x_i}. \quad (5.76)$$

We note that factor of two stems from the choice of viscous term since there are now two derivatives in $\mathcal{F}\boldsymbol{u}$, namely $\nabla\,\boldsymbol{u}$ and $(\nabla\,\boldsymbol{u})^T$, each one approximately commuting with the Laplacian term $\nabla\cdot\rho^{-1}\nabla$. As mentioned when discussing commutation of the viscous term in (5.56), the derivatives act in different directions and so, with the density and viscosity factors present, we do not have exact commutation except in each one-dimensional coordinate direction. Nonetheless, the number of derivatives on each factor remains correct. Moreover, the choice of convective term also maintains this property of having the correct number of derivatives acting on the factors, this time also on the wind $\boldsymbol{w}$, but we again do not commute exactly in higher dimensions. Furthermore, integration by parts in the weak form of the convective term is actually an application of the divergence theorem and does not assume anything on $\boldsymbol{w}$, that is, the error stems from the approximate commutator not the use of integration by parts (cf. Remark 6). To be explicit, the weak form of $\mathcal{F}_p$ is

$$\begin{aligned}\mathcal{F}_p(p,q) =& -2\int_\Omega\left(\nabla\cdot\left(\rho^{-1}\,\nabla\,p\right)\right)q+\int_\Omega\nabla\cdot\left(\boldsymbol{w}\,\mu^{-1}\,p\right)q+\Delta t^{-1}\int_\Omega\mu^{-1}\,p\,q\\ =& \;2\int_\Omega\rho^{-1}\,\nabla\,p\cdot\nabla\,q-\int_\Omega\mu^{-1}\,(\boldsymbol{w}\cdot\nabla\,q)\,p+\Delta t^{-1}\int_\Omega\mu^{-1}\,p\,q\\ &+\int_{\partial\Omega}\left(-2\,\rho^{-1}\frac{\partial p}{\partial n}+\mu^{-1}\,\boldsymbol{w}\cdot\boldsymbol{n}\,p\right)q,\end{aligned} \quad (5.77)$$

and setting the boundary term to zero yields the appropriate Robin condition, this time with the factor of two due to the form of the viscous term used.

Now that we have determined the appropriate bi-weighted commutator (5.74) and the corresponding $\mathcal{F}_p$ (5.75), we must construct the inverse Schur complement approximation. As in the single-phase case, we apply $\mathcal{F}^{-1}\mathcal{B}^T$ from the right to (5.74) and formally rearrange to yield

$$\left(\mathcal{B}\mathcal{F}^{-1}\mathcal{B}^T\right)^{-1}\approx(\mathcal{B}\rho^{-1}\mathcal{B}^T)^{-1}\mathcal{F}_p\mu. \quad (5.78)$$

When it comes to discretisation of (5.78) we have some choice. Firstly, in the spirit of the original single-phase PCD methodology the clearest path forward is to discretise as

$$\widehat{S}^{-1}=\left(A_p^{(1/\rho)}\right)^{-1}F_p\left(M_p^{(1/\mu)}\right)^{-1}. \quad (5.79)$$

Now the corresponding pressure space matrices are

$$F_{p;i,j} = 2\int_\Omega \rho^{-1}\,\nabla\,\psi_j\cdot\nabla\,\psi_i - \int_\Omega \mu^{-1}\,(\boldsymbol{w}\cdot\nabla\,\psi_i)\,\psi_j + \Delta t^{-1}\int_\Omega \mu^{-1}\,\psi_j\,\psi_i, \qquad (5.80)$$

with the Robin inflow condition already being incorporated, the inverse density scaled Laplacian (5.32) with boundary conditions as in (5.24), and the mass matrix $M_p^{(1/\mu)}$ as in (5.27) but without the factor of a half present. In addition, note that the Dirichlet conditions on the outflow boundary for both F_p and $A_p^{(1/\rho)}$ must be enforced explicitly. This formulation retains the benefits of the single-phase case, including the fact that both the diffusive and the mass terms built in F_p are the same as those required for the inverses on either side of F_p.

However, the inverse viscosity scaling within the convective and time-stepping terms might be viewed as unnatural and is, in fact, unnecessary, being an artefact of the full bi-weighted commutator. To see this we go back to the consideration of discretising the term $\mathcal{F}_p\mu$ in (5.78). Since this stands as a whole, there is no requirement to separate these two terms, except for the diffusive term in $\mathcal{F}_p$, as discussed previously. As such, we can reformulate this as

$$\mathcal{F}_p\mu = \mathcal{A}_p^{(1/\rho)}2\mu + \mathcal{N}_p + \mathcal{T}_p, \qquad (5.81)$$

where $\mathcal{A}_p^{(1/\rho)} = \mathcal{B}\rho^{-1}\mathcal{B}^T = \nabla\cdot\rho^{-1}\nabla$, $\mathcal{N}_p = \nabla\cdot\boldsymbol{w}$, and $\mathcal{T}_p = \Delta t^{-1}$. Discretising this form of the continuous operators yields

$$\widehat{S}^{-1} = \left(M^{(1/\mu)}\right)^{-1} + \left(A_p^{(1/\rho)}\right)^{-1}\left(N_p^{(1)} + \Delta t^{-1}M^{(1)}\right)\left(M^{(1)}\right)^{-1}, \qquad (5.82)$$

which we recognise as the *split* form of two-phase PCD previously derived in (5.37) and (5.64). Here, similar to (5.61a), the term $\mathcal{A}_p^{(1/\rho)}$ cancels for the diffusive part (splitting the boundary conditions as in Section 5.3.2) and $N_p^{(1)}$ is again given as (5.65) after integration by parts.

We see that the use of an appropriate *bi-weighted* approximate commutator is able to account for the full two-phase (variable coefficient) nature of our model problem. By maintaining all the analysis and manipulation in the continuous setting we gain valuable insight into the appropriate form of the operators as well as well boundary conditions on them. Further, we find that two forms of the resulting two-phase PCD preconditioner can result depending on the choice of how we discretise the inverse Schur complement approximation (5.78). Both forms are, in some sense, natural and we note the split form can be derived using weaker heuristics as presented in Sections 5.2 and 5.3.2. We will investigate any differences between these two formulations in our numerical results.

First, however, we must discuss the implementation of these preconditioners and, in particular, making them practical by using approximations of the inverses required. We will also discuss the case of utilising discontinuous pressure approximation through the example of $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ finite elements.

5.4 Implementation details

In this section we provide some additional practical details on implementing PCD-type preconditioners and particular choices we make in our numerical implementation using the software package IFISS [209].

5.4.1 Practical forms of the preconditioner

The Schur complement preconditioners discussed in the previous sections are "ideal" preconditioners, that is, it is assumed the inverses are computed exactly. On large problems, for a practical preconditioner the inverses required cannot be computed exactly (to within machine precision, such as using Gaussian elimination) but must be approximated by relatively cheap and efficient methods.

Within a PCD-type inverse Schur complement approximation we require inverses of a (scaled) Laplacian-type operator $A_p^{(\pi)}$ as well as a (scaled) mass matrix $M_p^{(1/\eta)}$, using the general scaling from (5.66). Note that in the split form of PCD (5.64) we require the inverse action of two different mass matrices.

For the inversion of Laplacian-type operators one of the best approaches available is to use a multigrid method [39]. Such methods can often deliver optimal (linear) complexity in terms of the number of unknowns for the solution of Poisson problems which require the inverse of a Laplacian-type operator. Multigrid methods are broadly split into two main classes: geometric multigrid and algebraic multigrid (AMG). The former utilises geometric information to build a hierarchy of grids while the latter uses solely algebraic information from the matrix representing the operator. While multigrid methods can be used to solve problems iteratively to a tolerance, our interest is using them to provide approximations to the action of the inverse. In this case it is preferable to have this action be fixed when formulating the preconditioner so that we can use GMRES as opposed to a flexible variant, required when the preconditioner is not fixed throughout the GMRES iterations. We utilise an AMG method which provides a Ruge and Stüben implementation, specifically the `HSL_MI20` code [29]. To provide a fixed approximation to the inverse of $A_p^{(\pi)}$ we use a single V-cycle, which is the default behaviour in IFISS [209]; see [71, Section 9.3.3] for more precise details.

For the solution of equations involving the Galerkin mass matrix there are typically two approaches used when the mass matrix is not already diagonal (this latter case occurs for the unscaled mass matrix when a discontinuous pressure approximation is used). First, it is well known that diagonal approximation of the mass matrix can yield good results [246, 71]. In particular, the diagonal matrix

$$D_p^{(1/\eta)} = \text{diag}\left(M_p^{(1/\eta)}\right) \tag{5.83}$$

is spectrally equivalent to $M_p^{(1/\eta)}$, that is

$$c_1\left(D_p^{(1/\eta)}\mathbf{p}, \mathbf{p}\right) \leq \left(M_p^{(1/\eta)}\mathbf{p}, \mathbf{p}\right) \leq c_2\left(D_p^{(1/\eta)}\mathbf{p}, \mathbf{p}\right) \quad \forall \mathbf{p} \in \mathbb{R}^{n_p}, \tag{5.84}$$

for some constants $c_1 > 0$ and $c_2 \geq c_1$ with $(\cdot, \cdot)$ being the standard l^2 inner product. Since the Galerkin mass matrix is symmetric positive definite, this means that all the eigenvalues λ of $(D_p^{(1/\eta)})^{-1}M_p^{(1/\eta)}$, or equivalently the generalised eigenvalues λ satisfying

$$\det\left(M_p^{(1/\eta)} - \lambda D_p^{(1/\eta)}\right), \tag{5.85}$$

are bounded as

$$c_1 \leq \lambda \leq c_2. \tag{5.86}$$

These bounds are typically independent of the mesh refinement, as with the bounds given in [246], hence the preconditioned operator $(D_p^{(1/\eta)})^{-1}M_p^{(1/\eta)}$ has a bounded spectrum independent of the mesh refinement and $D_p^{(1/\eta)}$ is a good preconditioner for $M_p^{(1/\eta)}$.

For use within a PCD-type Schur complement preconditioner we require a cheap (ideally fixed) approximation to the inverse of $M_p^{(1/\eta)}$ and this can be simply given by using the inverse of $D_p^{(1/\eta)}$ instead, as justified in the above spectral equivalence. We note that alternative forms of diagonal approximation to the mass matrix can also be used, for instance through lumping of the mass matrix. A standard lumping procedure is to sum the entries of each row of the mass matrix and put the corresponding values into a diagonal matrix $\bar{D}_p^{(1/\eta)}$. If we let $\mathbf{1}$ denote the appropriately sized vector of all ones, then the diagonal entries are given by

$$\bar{D}_{p;i,i}^{(1/\eta)} = \left(M_p^{(1/\eta)}\mathbf{1}\right)_i. \tag{5.87}$$

Again, the inverse of $\bar{D}_p^{(1/\eta)}$ can be used as an approximation to the inverse of the full mass matrix $M_p^{(1/\eta)}$. We do not explore the difference between lumping of the mass matrix and directly taking the diagonal entries of it.

While a diagonal approximation can be good, more accurate approximations can also be used. In [246], a conjugate gradient method with diagonal preconditioning is discussed for the solution of systems involving the Galerkin mass matrix. However, this gives an nonlinear operator for the Schur complement approximation. A more favourable approach which maintains linearity is the use of Chebyshev semi-iteration [81]. In particular, [248] consider a Chebyshev method with Jacobi splitting, namely using the diagonal of the mass matrix as a preconditioner for a stationary iterative method.

The Chebyshev semi-iterative method discussed in [248] allows us to obtain a better approximation of the inverse of $M_p^{(1/\eta)}$, however, it requires good bounds on the eigenvalues of the preconditioned operator. Fortunately, such bounds for the Galerkin mass matrix are known for common choices of finite elements [246]. These bounds are shown for the unscaled mass matrix preconditioned by its diagonal. In the case of a scaled mass matrix, the same bounds are appropriate since the diagonal includes the same scaling. As an example, the appropriate bounds we use for the $\boldsymbol{Q}_1$ mass matrix on a rectangular grid are

$$\frac{1}{4} \leq \lambda \leq \frac{9}{4}. \tag{5.88}$$

In our numerical results we will use three steps of Chebyshev semi-iteration as a fixed approximation to the inverse of any (non-diagonal) mass matrix required.

To give a practical inverse Schur complement approximation for the PCD-type preconditioners we have discussed, omitting the scaling to speak in general terms, we let $\widehat{A}_p$ denote the single AMG V-cycle approximation to the Laplacian term and $\widehat{M}_p$ denote the approximation of the mass matrix given by three steps of Chebyshev semi-iteration. Then, when no splitting is done, as in (5.79), we have

$$\widehat{S}^{-1} = \widehat{A}_p^{-1} F_p \widehat{M}_p^{-1}, \tag{5.89}$$

while for the split case we have

$$\widehat{S}^{-1} = \widetilde{M}_p^{-1} + \widehat{A}_p^{-1} \left(N_p + \Delta t^{-1} M_p \right) \widehat{M}_p^{-1}, \tag{5.90}$$

where $\widetilde{M}_p$ uses the same approximation as $\widehat{M}_p$ but with a (potentially) different scaling, as in (5.64). Note that, in this latter form, $\widetilde{M}_p$ need not use the same type of approximation as $\widehat{M}_p$; we do not explore this observation but note that tuning of these approximations, including the multigrid approach for $\widehat{A}_p$, could be further investigated and refined.

While the practical form (5.89) is successfully used in [71, 209], we note that one feature of the underlying continuous operators is lost. Namely, due the approximate inverses $\widehat{A}_p^{-1}$ and $\widehat{M}_p^{-1}$, the implicitly included terms $\widehat{A}_p^{-1}A_p$ and $M_p\widehat{M}_p^{-1}$ no longer simplify to the identity. We see, using the splitting $F_p = A_p + N_p + \Delta t^{-1}M_p$, that we now have

$$\widehat{S}^{-1} = \widehat{A}_p^{-1}A_p\widehat{M}_p^{-1} + \widehat{A}_p^{-1}N_p\widehat{M}_p^{-1} + \Delta t^{-1}\widehat{A}_p^{-1}M_p\widehat{M}_p^{-1}. \tag{5.91}$$

A form which more closely matches the cancellation of the operators which occurs for the underlying continuous operators and in the ideal case would be

$$\widehat{S}^{-1} = \widetilde{M}_p^{-1} + \widehat{A}_p^{-1}N_p\widehat{M}_p^{-1} + \Delta t^{-1}\widehat{A}_p^{-1}, \tag{5.92}$$

where we allow that $\widetilde{M}_p$ may be different, for instance in scaling, to $\widehat{M}_p$ and so incorporate the split form similar to (5.90).

It is not immediately clear whether implementation via (5.92) or as (5.89)–(5.90) is preferable. Whilst (5.92) avoids the approximations not cancelling, a potential benefit of the forms (5.89)–(5.90) is that the error in the practical approximations apply consistently throughout all terms of $\widehat{S}^{-1}$. For the approximation of the time-stepping term, we find that there is relatively little difference between the two approaches (we will see this in the results of Table 5.2 in comparison with those in Table A.1). The largest difference is seen for the case of small time-steps and large Reynolds number (precisely when the time-stepping term is most dominant), in this case having the best approximation for this term alone, as in (5.92), is favourable. Outside of this case, including when the Reynolds number as well as the time-step is small, the consistently approximated forms (5.89)–(5.90) tend to be slightly favourable. Since we are interested in the large Reynolds number and small time-step case and since it gives the largest difference, we use the form (5.92), which is most favourable here.

Note that for the viscosity scaled PCD approach (5.50) the scaling in the two mass matrices, one from the time-stepping term in F_p and the other stemming from the viscosity weighted commutator, are different. Since the scaling does not match, we can no longer cancel these terms. As such we always use the form (5.90) for the viscosity scaled PCD approach. We see that in using a form such as (5.92) this issue could be corrected by using the appropriate inverse density scaled Laplacian for the final time-stepping term in (5.92), however, we then would require two distinct Laplacian solves and thus twice the set-up time for the AMG approximation. We will see that such additional work is unnecessary and better performance is obtained from the other variants of two-phase PCD, as such we do not consider this issue further.

Considering now the approximation of the diffusive term and $\widehat{A}_p^{-1}A_p$, we find little difference between the two approaches in (5.89) and (5.92) for inflow–outflow problems. Note that in the split form (5.92) we make use of our notion that integration by parts for the pressure space convective term N_p to allow us to split F_p and cancel the Laplacian terms. Given that little difference between (5.89) and (5.92) is observed for inflow–outflow problems this validates our approach to the boundary conditions when utilising a split form of PCD. However, in the case of enclosed flow we observe a difference between utilising the form (5.89) and the split form (5.92). We believe this is due to the way the singular operator A_p is handled rather than an inherent issue with using a form involving $\widehat{A}_p^{-1}A_p$, as we now describe.

For enclosed flow the Laplacian matrix A_p is singular since a Neumann condition is imposed everywhere on the boundary and so the matrix has a null space of dimension one corresponding to the vector of all ones, denoted $\mathbf{1}$. The discussion in [71, Section 9.3.5] shows that using such singular operators causes no difficulty for the iterative solver. When we apply A_p^{-1} to a vector $\mathbf{q}$ we must ensure that we take out the null vector $\mathbf{1}$ from $\mathbf{q}$ so that we can find a solution to $A_p\mathbf{q}^\star = \mathbf{q}$. This is done by making use of the projection

$$P_{\mathbf{1}} = I - \frac{\mathbf{1}\mathbf{1}^T}{\|\mathbf{1}\|_2^2}, \tag{5.93}$$

that is, we compute $\mathbf{q}^\star = A_p^{-1}P_{\mathbf{1}}\mathbf{q}$. To do so in a stable manner, IFISS calculates $P_{\mathbf{1}}\mathbf{q}$ then solves the reduced system formed by removing a row (and column for A_p) of $\mathbf{q}^\star A_p = P_{\mathbf{1}}\mathbf{q}$ corresponding to an index i_{null}, the remaining component of $\mathbf{q}^\star$ being set to zero. In the case of using a multigrid approximation $\widehat{A}_p^{-1}$ we instead solve the similarly reduced system corresponding to $\mathbf{q}^\star\widehat{A}_p = P_{\mathbf{1}}\mathbf{q}$. Since the Laplacian solve is the last operation in the inverse Schur complement approximation (5.89) (as well as in the ideal case (5.79)), the resulting vector $\mathbf{p}^\star = \widehat{S}^{-1}\mathbf{p}$ has a zero in the component corresponding to i_{null}. This is in contrast to using the split form (5.92) where now $\mathbf{p}^\star = \widehat{S}^{-1}\mathbf{p} = \widetilde{M}_p^{-1}\mathbf{p} + \mathbf{q}^\star$ where $\mathbf{q}^\star$ has a zero in the component corresponding to i_{null}. We see here that $\widetilde{M}_p^{-1}\mathbf{p}$ and thus $\mathbf{p}^\star$ generally do not have a zero in the component corresponding to i_{null}. Again, we emphasize that this is also true for the ideal variants and so the discrepancy between these two approaches does not stem from the multigrid approximation to A_p^{-1}. We have found that implementation via the split form (5.92) allows for faster convergence in the case of enclosed flow (as we shall exhibit in Table 5.3) and thus this form is preferred for Problem 1, the cavity problem. For Problem 2, the step problem, we do not have enclosed form so we can use (5.89).

As well as an approximation to the Schur complement, a crucial part of block triangular preconditioners of the form (4.11) is the approximation of the F block. For Navier–Stokes flow this represents a convection–diffusion operator on the velocity space. Typically multilevel approaches such as multigrid or domain decomposition are used to provide the approximation required in the action of $\widehat{F}^{-1}$. For single-phase flow, an appropriate algebraic multigrid method is included within IFISS, namely making use of the `HSL_MI20` code [29]. In particular, good performance is seen in the results of [71] where a single V-cycle is used to provide an approximation. As detailed in [71, Section 9.3.3], the key to providing robustness for the convection–diffusion operator lies in a hybrid smoothing strategy within the AMG solver. This applies a single sweep of $ILU(0)$ using a left to right, bottom to top ordering on the finest level while all coarser levels in the V-cycle make use of two presmoothing and two postsmoothing steps using a point-damped Jacobi smoother with an appropriate damping parameter.

However, we have found that this approach is not sufficiently robust to display good behaviour when considering the convection–diffusion operator arising in our model of two-phase flow. The primary additional difficulties stems from the highly varying and potentially discontinuous viscosity and density coefficients and the fact that the viscous operator no longer simplifies to a vector Laplacian term and contains non-zero off-diagonal sub-blocks. While we have investigated using only the block diagonal part of F to construct the AMG approximation, our experience suggests that it is more challenging to obtain robust behaviour of an AMG solver for the convection–diffusion operator using off-the-shelf methods. Nonetheless, there is a variety of work on more specialist approaches using multigrid or domain decomposition for models which incorporate highly varying coefficients, for instance in [84, 85, 199, 200]. This suggests it is likely that good methods can be constructed which are able to provide an efficient and robust approximation in the action of $\widehat{F}^{-1}$ required by block triangular preconditioners of the form (4.11). Note that, in principle, any such method can be used for the block preconditioners we consider.

Nevertheless, since our primary focus has been on appropriate Schur complement approximations, we have not extensively pursued this issue; further investigation on this topic is an important direction of future work. In order to assess the raw potential of the block preconditioners we have derived we will use a direct solver to give the action of $\widehat{F}^{-1}$ throughout this work, including in the LSC approaches considered in the next chapter. This approach will not yield an efficient preconditioner for large-scale problems but is feasible for the problems that we consider.

5.4.2 Discontinuous pressure approximation

We now discuss the case of discontinuous pressure approximation, using the example of $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ finite elements. Recall that in order to define F_p in (5.11) based on the continuous operator $\mathcal{F}_p$ we made the assumption that the pressure approximation was continuous. This assumption allows us to construct a pressure Laplacian, as well as a pressure space convection term, since we can take appropriate weak derivatives of the pressure p. If the pressure is discontinuous we can no longer take these weak derivatives and more work is required to construct appropriate discrete operators.

To construct discrete operators when the pressure is discontinuous across elements, we do so locally on elements and utilise edge contributions. The construction is more involved with the main ideas being detailed in [71, Section 9.2.1] and a proof of concept implementation being found within IFISS [209]. We note that the derivation is done assuming a $\boldsymbol{P}_0$ approximation of the pressure. As such, when using $\boldsymbol{P}_{-1}$ we discard the local slope information and so build operators based on a single value of the pressure at the centroid of each element. Since we only require approximations to the operators within the preconditioner, this suffices to be effective and provides good results [71, Remark 9.4]. We note that the resulting approach has a resemblance to discontinuous Galerkin (DG) type formulations (see, for example, [5]).

To describe how the discrete operators are constructed for two-phase PCD we follow the exposition in [71, Section 9.2.1] but note when our arguments differ and what additional assumptions are made in the two-phase case. We start by considering a discrete form for $\mathcal{N}_p^{(1/\mu)}$, the convective term in (5.75), and note that the same follows for the unscaled term $\mathcal{N}_p^{(1)}$ in (5.61b) by simply setting $\mu \equiv 1$. We remark that we cannot start from the form after integration by parts since now boundary terms would occur on element edges. In fact, it is these edge contributions that allow us to build an appropriate discrete operator.

We assume a $\boldsymbol{P}_0$ approximation of the pressure p_h corresponding to the discrete coefficient vector $\mathbf{p}$. To derive an expression for $N_p^{(1/\mu)}$, we first consider a (formal) weak form $\mathcal{N}_p^{(1/\mu)}(p_h, q)$ where $q = \psi_i$ is the $\boldsymbol{P}_0$ basis function with support only in element T_i. That is, ψ_i is the pressure basis function which is a constant unit pressure over T_i but zero elsewhere. Now we must break up the integral over Ω to integrate elementwise over all elements in $\mathcal{T}_h$ giving

$$\mathcal{N}_p^{(1/\mu)}(p_h, \psi_i) = \sum_{T \in \mathcal{T}_h} \int_T \nabla \cdot \left(\boldsymbol{w}\, \mu^{-1}\, p_h \right) \psi_i = \int_{T_i} \nabla \cdot \left(\boldsymbol{w}\, \mu^{-1}\, p_h \right), \tag{5.94}$$

since $\psi_i \equiv 1$ in T_i and is only supported in T_i.

We now utilise integration by parts (in this case equivalent to the divergence theorem) to obtain

$$\mathcal{N}_p^{(1/\mu)}(p_h, \psi_i) = \sum_{E \in \mathcal{E}(T_i)} \int_E \mu^{-1} p_h \, \boldsymbol{w} \cdot \boldsymbol{n}_{E,T_i}, \tag{5.95}$$

where $\mathcal{E}(T_i)$ denotes the set of edges of the element T_i and $\boldsymbol{n}_{E,T_i}$ is the outward pointing normal on E with respect to element T_i. We note that the argument differs from [71] since our form of $\mathcal{N}_p$ is the divergence of a function and so we can directly apply the divergence theorem to yield edge contributions while [71] starts from

$$\int_{T_i} \boldsymbol{w} \cdot \nabla \, p_h = - \int_{T_i} p_h \, \nabla \cdot \boldsymbol{w} + \sum_{E \in \mathcal{E}(T_i)} \int_E p_h \, \boldsymbol{w} \cdot \boldsymbol{n}_{E,T_i}, \tag{5.96}$$

with the element integrals on the right-hand side being zero since the discretisation provides elementwise conservation of mass. While the two arguments provide the same approximation in the single-phase case, the latter does not apply in the two-phase case since now the element integral on the right-hand side of (5.96) would include a scaling with μ^{-1} and so elementwise conservation of mass can no longer be used to set this term to zero. The different starting points stems from whether we have passed $\boldsymbol{w}$ through the derivative or not and so we see there is an imperative reason in the two-phase case not to do so since otherwise we cannot formulate the pressure space convective term solely in terms of edge contributions. That is, we must utilise the form $\mathcal{N}_p^{(1/\mu)} p = \nabla \cdot (\boldsymbol{w} \mu^{-1} p)$ and not $\mathcal{N}_p^{(1/\mu)} p = (\boldsymbol{w} \cdot \nabla \,) \mu^{-1} p$; see the discussion in Section 5.3.3.

We see from (5.95) that the contribution in the weak form from ψ_i is precisely the edge contribution along the edges of the element T_i and thus depends only on the ith component of $\mathbf{p}$. As such, this would yield a diagonal approximation for the convective term. In order to gain a better approximation we utilise equidistribution of the edge contributions. We note that each edge contributes to two elements (aside from those on the boundary $\partial\Omega$). We can equidistribute these contributions between the two elements sharing the edge by introducing the directional jump operator

$$[\![\mu^{-1} p_h \, \boldsymbol{w} \cdot \boldsymbol{n}]\!]_{E,T} = \left(\mu^{-1} p_h|_T - \mu^{-1} p_h|_{T_E} \right) \boldsymbol{w} \cdot \boldsymbol{n}_{E,T} = [\![\mu^{-1} p_h]\!]_{E,T} \, \boldsymbol{w} \cdot \boldsymbol{n}_{E,T}, \tag{5.97}$$

where T and T_E are the elements sharing the edge E. We equidistribute using the fact that, aside from suitable modification for edges on the boundary,

$$\sum_{T \in \mathcal{T}_h} \sum_{E \in \mathcal{E}(T)} \int_E \mu^{-1} p_h \, \boldsymbol{w} \cdot \boldsymbol{n}_{E,T} = \frac{1}{2} \sum_{T \in \mathcal{T}_h} \sum_{E \in \mathcal{E}(T)} \int_E [\![\mu^{-1} p_h \, \boldsymbol{w} \cdot \boldsymbol{n}]\!]_{E,T}. \tag{5.98}$$

Thus we use the weak form defined by

$$\mathcal{N}_p^{(1/\mu)}(p_h, \psi_i) = \frac{1}{2} \sum_{E \in \mathcal{E}(T_i)} \int_E [\![\mu^{-1} \, p_h \, \boldsymbol{w} \cdot \boldsymbol{n}]\!]_{E,T_i}. \tag{5.99}$$

We can further approximate the edge integrals through an evaluation at the midpoint of each edge, denoted $\boldsymbol{x}_E$ for edge E, and obtain

$$\mathcal{N}_p^{(1/\mu)}(p_h, \psi_i) = \frac{1}{2} \sum_{E \in \mathcal{E}(T_i)} h_E [\![\mu^{-1} \, p_h]\!]_{E,T_i} \, \boldsymbol{w} \cdot \boldsymbol{n}_{E,T_i}(\boldsymbol{x}_E), \tag{5.100}$$

where h_E is the length of edge E. Here, if we assume that μ is piecewise constant across the edges, as is true for p_h, this approximation is exact for $\boldsymbol{Q}_1$–$\boldsymbol{P}_0$ elements since $\boldsymbol{w} \cdot \boldsymbol{n}_{E,T_i}$ is a linear function on the edge E.

For simplicity in our exposition we make the aforementioned assumption, namely that the interface lies along edges of the elements used and so the viscosity on either side of an edge is constant. We consider the case of grid-aligned square elements with side length h (as we will use in IFISS). For the approximation of the ith component of $N_p^{(1/\mu)}\mathbf{p}$, we suppose that the ith component of the discrete pressure $\mathbf{p}$ is given by the value p_0 and use the subscripts $\pm x$ and $\pm y$ to denote values from the neighbouring elements in the positive or negative x or y directions respectively. For example, p_{+x} refers to the pressure degree of freedom at the centroid of the neighbouring element in the positive x direction while μ_{+x} refers to the viscosity in this element. Further, in a similar manner, we let $\boldsymbol{x}_{\pm x}$ and $\boldsymbol{x}_{\pm y}$ be the midpoints along the edges with the neighbouring elements. With this notation we then can write our approximation to $N_p^{(1/\mu)}$ as

$$\begin{aligned} \left[N_p^{(1/\mu)}\mathbf{p}\right]_i = \quad & \frac{h}{2}\left(\mu_0^{-1}\mathrm{p}_0 - \mu_{-x}^{-1}\mathrm{p}_{-x}\right) w_x(\boldsymbol{x}_{-x}) - \frac{h}{2}\left(\mu_0^{-1}\mathrm{p}_0 - \mu_{+x}^{-1}\mathrm{p}_{+x}\right) w_x(\boldsymbol{x}_{+x}) \\ & + \frac{h}{2}\left(\mu_0^{-1}\mathrm{p}_0 - \mu_{-y}^{-1}\mathrm{p}_{-y}\right) w_y(\boldsymbol{x}_{-y}) - \frac{h}{2}\left(\mu_0^{-1}\mathrm{p}_0 - \mu_{+y}^{-1}\mathrm{p}_{+y}\right) w_y(\boldsymbol{x}_{+y}) \end{aligned} \tag{5.101}$$

with suitable modification for elements on the boundary and where w_x and w_y are the x and y component of the wind $\boldsymbol{w}$ respectively. It is noted in [71] that if the wind $\boldsymbol{w}$ was constant then this approximation is equivalent to a standard centred finite difference approximation defined at the centroids of the elements. Namely, since the p_0 terms vanish, we have the approximation

$$\nabla \cdot \left(\boldsymbol{w}\mu^{-1}p_h\right)\big|_{T_i} \approx \frac{\mu_{+x}^{-1}\mathrm{p}_{+x} - \mu_{-x}^{-1}\mathrm{p}_{-x}}{2h} w_x + \frac{\mu_{+y}^{-1}\mathrm{p}_{+y} - \mu_{-y}^{-1}\mathrm{p}_{-y}}{2h} w_y, \tag{5.102}$$

where we have also divided by the element area h^2 to give the correct scaling compared to (5.101).

Now consider the diffusive term $\mathcal{A}_p^{(1/\rho)}$ required. We again follow [71] for grid aligned square elements and build a discrete operator using centred approximations to the second derivatives, the only difference being the inverse density scaling now present. In such approximations we require the value of the diffusion coefficient, here ρ^{-1}, at the midpoints of the element edges. For this we have found it beneficial to take an arithmetic average of the two density values in the midpoint contribution when density differs across an edge. Hence, using notation as above for the convective term, the corresponding approximation to the Laplacian term for the degree of freedom corresponding to element T_i is given by (cf. [71, equations (9.29)–(9.30)])

$$\begin{aligned}\nabla \cdot \left(\rho^{-1} \nabla p_h\right)\big|_{T_i} \approx \quad & \frac{1}{h}\left(\frac{2}{\rho_{+x}+\rho_0}\frac{\mathrm{p}_{+x}-\mathrm{p}_0}{h} - \frac{2}{\rho_0+\rho_{-x}}\frac{\mathrm{p}_0-\mathrm{p}_{-x}}{h}\right) \\ & + \frac{1}{h}\left(\frac{2}{\rho_{+y}+\rho_0}\frac{\mathrm{p}_{+y}-\mathrm{p}_0}{h} - \frac{2}{\rho_0+\rho_{-y}}\frac{\mathrm{p}_0-\mathrm{p}_{-y}}{h}\right).\end{aligned} \tag{5.103}$$

Again we have assumed for simplicity that the interface lies along edges of the elements used. To give the corresponding expression for the discrete Laplacian $A_p^{(1/\rho)}$ we simply need to multiply the expression (5.103) by the element area h^2.

Finally we require the relevant (scaled) mass matrices. These can be constructed in the usual manner since no derivatives are required. Since the $\boldsymbol{P}_0$ basis function are supported on distinct elements the resulting mass matrix is always diagonal. However, we are interested in the use of $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ elements. Here we have two additional basis functions corresponding to the x and y slope of the pressure for each element. Nonetheless, the resulting mass matrix is still diagonal so long as the scaling factor, for instance μ^{-1}, is constant on each element, which is true when interface lies along element edges and no smoothing of the density or viscosity fields is used. However, even when the scaling factor varies within an element the mass matrix obtained is still block diagonal with block size three, corresponding to the three basis functions with support in any given element. As such the mass matrix is still easily inverted and thus no diagonal approximation or Chebyshev semi-iteration is required in the case of discontinuous pressure approximation. Returning to the matrices $N_p^{(1/\mu)}$ and $A_p^{(1/\rho)}$ for $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ elements, the construction above for $\boldsymbol{P}_0$ elements is used for the degrees of freedom corresponding to the element centroid values. For the degrees of freedom corresponding to the local slopes we ignore these for $N_p^{(1/\mu)}$ (that is, we have zero rows and columns) while for $A_p^{(1/\rho)}$ we simply scale the input values commensurately by ρ^{-1} (that is the elementwise values for ρ^{-1} are put onto the diagonal). We reiterate that these operators need only define suitable approximations for use within the preconditioner and, as we shall see, the resulting approach works very well.

5.5 Illustrative numerical results

5.5.1 The ideal single-phase case and implementation of the Robin condition

Before addressing the two-phase problem we provide results for the single-phase case using the ideal form of the PCD preconditioner (5.9). This gives an example of the best performance we can hope for and allows us to see what, if anything, changes in the behaviour of PCD when applied to an example of two-phase flow. We also use this as a chance to compare our *natural* implementation of the Robin condition on the inflow boundary, using integration by parts, with the implementation currently in IFISS which requires this condition to be explicitly enforced; see Remark 6.

In order to test the Robin condition we use a steady inflow–outflow problem, namely the step problem as described in Problem 2a. We use square $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ elements having side length h and consider Picard iteration. In Table 5.1 we provide GMRES iteration counts for the second Picard iteration; similar iteration counts are seen at other Picard iterations. For both implementations of the Robin condition we see effectively mesh-independent convergence and only a mild dependence on the Reynolds number (this dependence is demonstrated to be $\mathcal{O}(Re^{1/2})$ for the steady problem in [138, 139]). Further, we observe that our new natural enforcement of the Robin boundary condition provides an improvement to the explicitly enforced implementation. In particular, for larger Reynolds numbers the reduction in iteration counts reaches up to 20%. We expect that, as the mesh is refined, the iteration counts recorded by the two approaches becomes closer since the explicit approach is better able to enforce the Robin condition; we begin to see this by the final row of Table 5.1.

Table 5.1: Preconditioned GMRES iterations using single-phase PCD (5.9) with the Robin condition enforced explicitly / naturally (through integration by parts) for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady backward-facing step problem (Problem 2a) using Picard iteration with varying Reynolds number Re and grid size h.

$h \setminus Re$	10	100	200
$\frac{1}{4}$	20 / 18	27 / 22	37 / 33
$\frac{1}{8}$	20 / 18	25 / 20	29 / 24
$\frac{1}{16}$	19 / 19	25 / 21	30 / 24
$\frac{1}{32}$	19 / 19	24 / 22	29 / 25
$\frac{1}{64}$	18 / 18	23 / 22	27 / 25

5.5.2 A comparison of two-phase PCD variants

To give an illustrative comparison of the two-phase PCD variants we have considered and how well they perform we again consider a step problem, this time that described in Problem 2 and now in its two-phase air–water variant. As in the single-phase case, we use square $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ elements but now consider the more challenging case of Newton linearisation. We will vary the grid parameter h, the Reynolds number Re, and the time-step Δt in the time-dependent case. To give an indication of the performance we tabulate the average number of GMRES iterations required over the course of Newton's method to reach a nonlinear tolerance of 10^{-10}; we omit the initial Stokes solve that provides the initial guess and the average values stated have been rounded.

Table 5.2 displays results for the inflow–outflow step problem using the viscosity scaled formulation (5.30), two-phase PCD (5.79) derived through the bi-weighted commutator and in the split form (5.64). In addition, we also tabulate results for the split form without any adjustments for boundary conditions as considered in (5.37). Firstly, we see that all formulations which take into account appropriate boundary conditions provide effectively mesh-independent convergence. This includes the viscosity scaled formulation, even in the time-dependent case where we know that the resulting approximation does not default to the correct Cahouet–Chabard preconditioner. Nonetheless, we see that the overall performance of this viscosity scaled approach is poorer in the time-dependent case and as the time-step decreases the iteration counts tend not to decease as we might expect and as is observed for the other formulations. Moreover, even in the steady case the viscosity scaled approach gives iterations counts significantly higher than the other formulations, justifying the need for two different scalings to be incorporated.

Comparing two-phase PCD in the two different discretisations, one requiring the split form, we see very similar performance. While neither uniformly improves on the other we note that the split form requires slightly higher iterations counts for the larger Reynolds number flows on coarse meshes but on the finest mesh performance is almost identical. Such similar results validates our choice of boundary condition on the pressure space convection operator N_p in the split form. Furthermore, the importance of enforcing appropriate boundary conditions is seen in Table 5.2 by comparing the split two-phase PCD method with and without boundary adjustments: without appropriate conditions the iteration counts are much higher and not scalable.

For both variants of two-phase PCD we see a mild dependence on the Reynolds number for the steady problem, as in the singe-phase case, while for the time-dependent problem there is little dependence on the Reynolds number (cf. [138]).

Table 5.2: Average preconditioned GMRES iterations after Newton linearisation when using viscosity scaled PCD (5.50), two-phase PCD (5.79) derived using a bi-weighted commutator, split two-phase PCD (5.64), and split two-phase PCD without incorporating appropriate boundary conditions (as in Section 5.2) for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the backward-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow) and varying grid size h, Reynolds number Re (of the water) and time-step Δt.

		Viscosity scaled PCD (5.50)			Two-phase PCD (5.79)			Split two-phase PCD (5.64)			Split two-phase PCD without BCs		
Δt	$h \setminus Re$	10	100	500	10	100	500	10	100	500	10	100	500
$\frac{1}{10}$	$\frac{1}{4}$	22	41	64	15	14	20	15	14	21	23	23	30
	$\frac{1}{8}$	23	33	67	14	13	17	14	13	19	24	27	35
	$\frac{1}{16}$	23	33	58	15	12	16	15	12	18	30	36	45
	$\frac{1}{32}$	24	31	59	16	12	14	16	12	15	39	53	63
	$\frac{1}{64}$	23	30	53	17	13	13	17	13	14	56	80	98
1	$\frac{1}{4}$	23	30	51	21	19	22	21	20	25	26	29	36
	$\frac{1}{8}$	24	29	51	20	16	20	20	16	21	27	30	37
	$\frac{1}{16}$	24	30	55	21	17	17	21	17	18	31	39	45
	$\frac{1}{32}$	24	29	53	22	18	16	22	18	16	40	56	61
	$\frac{1}{64}$	24	29	47	22	19	16	22	19	16	57	82	95
10	$\frac{1}{4}$	24	32	62	23	24	32	23	25	36	28	35	46
	$\frac{1}{8}$	24	31	68	23	23	27	23	23	28	29	36	47
	$\frac{1}{16}$	25	33	60	24	24	24	24	24	24	33	45	52
	$\frac{1}{32}$	25	33	56	24	26	24	24	25	24	41	62	70
	$\frac{1}{64}$	25	33	56	24	26	25	24	26	26	57	91	104
Steady	$\frac{1}{4}$	23	36	95	23	27	58	23	27	63	28	37	78
	$\frac{1}{8}$	24	36	110	23	26	52	23	26	53	29	40	74
	$\frac{1}{16}$	25	39	91	24	27	47	24	27	47	33	48	78
	$\frac{1}{32}$	26	42	83	25	28	46	25	28	46	41	65	98
	$\frac{1}{64}$	26	41	85	25	30	48	25	29	48	58	98	135

We now consider the cavity problem described in Problem 1 and this time utilise $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ elements, allowing us to verify the construction detailed in Section 5.4.2 for discontinuous pressure approximation. We suppose Picard linearisation is employed on the steady problem and again tabulate the average number of preconditioned GMRES iterations required over the nonlinear iteration.

Table 5.3 provides results for the two-phase PCD (5.79) approach using the two different practical implementations discussed, namely via (5.92) where F_p has been split (and the diffusive terms cancelled) and (5.89) where it has not. Crucially, the cavity problem is a example of enclosed flow and so our discussion at the end of Section 5.4.1 regarding the singular Laplacian operator applies. Our results make it clear that the split implementation is much more favourable. In fact, without splitting the implementation (5.89) requires up to 50% more iterations and 30% on average. Further, while the split implementation shows very good scalability as the mesh is refined, without splitting a slight mesh dependence begins to show. With regards to the dependence on Reynolds number, we again see only a weak dependence by the time we reach the finest mesh used. We note that larger iteration counts are seen for large Reynolds number flows on coarse meshes; we attribute this to the fact that these flows are not well resolved on such meshes and that the discrete operators are less able to capture their continuous counterparts. Finally we note that, in this instance, results for the split two-phase PCD method (5.64) are identical to the split implementation of (5.79) and so the results are not included separately here. This again suggests that the inverse viscosity scaling within the pressure space convective term is not strictly necessary, as suggested in our analysis of the bi-weighted commutator in Section 5.3.3.

Table 5.3: Average preconditioned GMRES iterations after Picard linearisation when using two-phase PCD (5.79) implemented via (5.92) / (5.89) for the $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ solution to the steady lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h. Results for split two-phase PCD (5.64) are identical to the split implementation (5.92) and thus are not included separately here.

$h \setminus Re$	10	$10^{1.5}$	100	$10^{2.5}$	1000
$\frac{1}{8}$	17 / 23	21 / 26	27 / 33	41 / 48	88 / 100
$\frac{1}{16}$	17 / 24	21 / 27	27 / 33	38 / 47	73 / 86
$\frac{1}{32}$	18 / 25	21 / 28	27 / 34	34 / 42	55 / 71
$\frac{1}{64}$	18 / 26	21 / 30	27 / 36	35 / 44	40 / 50
$\frac{1}{128}$	18 / 27	21 / 31	27 / 37	34 / 45	42 / 52

Chapter 6

Least-squares commutator preconditioning

In this chapter we detail another commutator based approach to approximating the Schur complement and explore how it can be adapted to work with the variable coefficient nature of the equations describing two-phase flow. After presenting the main ideas in the single-phase case and discussing the role of boundary conditions, we first consider the simpler case of Stokes flow where some progress has already been made in applying similar methodology to the variable viscosity Stokes problem. We investigate an alternative formulation which can offer improvements in the case of two-phase flow, in particular for large viscosity ratios. Following this, we focus our attention on the more challenging Navier–Stokes problem. Here, it is less clear which scaling is most appropriate and it depends on the choice of norm used to formulate the least-squares problem. We present a numerical study and also consider issues such as smoothing of the interface and the imposition of boundary condition modifications.

6.1 The single-phase LSC preconditioner

The least-squares commutator (LSC) preconditioner was first introduced in [63] and later extended for use with stabilised finite elements ($C \neq 0$) in [65]. A similar form of preconditioner, called BFBt, was constructed earlier in [67] via different arguments and can be seen as an unscaled version of LSC. Indeed, it has subsequently been uncovered that the performance of such LSC-type methods depends crucially on the scaling incorporated. We shall see that this is particularly true in the more challenging variable coefficient problems we study. However, we first consider the single-phase case, showing how the method can be derived, extended to stabilised elements, and appropriate modifications for boundary conditions incorporated.

6.1.1 Derivation of the LSC preconditioner

As with the pressure convection–diffusion (PCD) approach, the derivation of the least-squares commutator (LSC) preconditioner starts with the notion of a commutator between the divergence $\mathcal{B}$ and convection–diffusion operator $\mathcal{F}$, namely

$$\mathcal{E} = \mathcal{B}\mathcal{F} - \mathcal{F}_p\mathcal{B}. \tag{6.1}$$

However, a key difference in the LSC approach is that we immediately discretise this commutator to consider the discrete commutator

$$\mathcal{E}_h = \left(M_p^{-1}B\right)\left(M^{-1}F\right) - \left(M_p^{-1}F_p\right)\left(M_p^{-1}B\right), \tag{6.2}$$

where F_p represents the discrete form of $\mathcal{F}_p$. We recall that M is the velocity mass matrix while M_p is the pressure mass matrix; they feature here to provide the correct scaling for the discrete operators. To avoid issues of appropriate stabilisation, we assume for now that an *inf–sup* stable discretisation is used with $C = 0$ so that the relevant Schur complement is given by $S = BF^{-1}B^T$.

We now seek F_p such that the commutator is small, as we did when seeking $\mathcal{F}_p$ for the PCD approach. However, for the LSC method this is formulated in a particular sense as we shall describe shortly. On equating $\mathcal{E}_h$ to be approximately zero, (6.2) can be rearranged to give the approximation

$$BF^{-1}B^T \approx M_pF_p^{-1}BM^{-1}B^T. \tag{6.3}$$

The manipulation is similar to the formal manipulation used for the continuous case in (5.2). By taking the inverse of (6.3) we obtain the required approximation to the inverse Schur complement

$$\left(BF^{-1}B^T\right)^{-1} \approx \left(BM^{-1}B^T\right)^{-1} F_pM_p^{-1}. \tag{6.4}$$

Note that this provides a different discretisation of the (scaled) Laplacian term $\mathcal{B}\mathcal{B}^T$ in the approximation compared to that used by PCD, namely A_p in (5.8).

In order to find F_p (a matrix) from the discrete commutator (6.2), a least-squares problem is formulated to minimise the commutator. There are several equivalent viewpoints on how this can be done. Firstly, we note that the discrete commutator (6.2) is a matrix and so we must consider a matrix norm. A convenient norm for such a least-squares problem is the (weighted) Frobenius norm $\|\cdot\|_{F,M^{-1}}$, where the weighting M^{-1} gives the appropriate scaling in the finite element context. For a positive definite weight matrix H, this norm is defined (see, for instance, [106]) by

$$\|A\|_{F,H}^2 = \operatorname{trace}(AHA^T). \tag{6.5}$$

Now that we have chosen an appropriate norm, minimising the commutator in this norm leads to the weighted least-squares problem

$$\min_{M_p^{-1}F_p} \left\| M_p^{-1}BM^{-1}F - M_p^{-1}F_pM_p^{-1}B \right\|_{F,M^{-1}}. \tag{6.6}$$

This problem can be solved by writing the associated normal equations

$$M_p^{-1}BM^{-1}FM^{-1}B^TM_p^{-1} = M_p^{-1}F_pM_p^{-1}BM^{-1}B^TM_p^{-1}. \tag{6.7}$$

A straightforward rearrangement reveals that

$$F_p = BM^{-1}FM^{-1}B^T\left(BM^{-1}B^T\right)^{-1}M_p \tag{6.8}$$

and thus the inverse Schur complement approximation

$$\left(BF^{-1}B^T\right)^{-1} \approx \left(BM^{-1}B^T\right)^{-1}BM^{-1}FM^{-1}B^T\left(BM^{-1}B^T\right)^{-1}. \tag{6.9}$$

We note that this approximation does not require the construction of additional matrices on the pressure space since $F_pM_p^{-1}$ is defined in terms of matrices stemming directly from the saddle point system, except in the steady case where the velocity mass matrix M does not already feature. Further, as compared with PCD, we now have two Laplacian terms and they are both given directly by $BM^{-1}B^T$.

An alternative viewpoint, described in [71], is to minimise a discretisation of the adjoint of the commutator through the consideration of individual columns of F_p^T and an appropriately weighted norm on discrete velocity space. The choice of the adjoint, and so the transpose of F_p, is made so that the weighted least-squares problem arises in a more succinct and straightforward manner. Note that the adjoint problem involves commutation with the gradient as opposed to the divergence. The resulting set of weighted least-squares problems used in [71] for the columns $\left[F_p^T\right]_j$ are

$$\min_{\left[F_p^T\right]_j} \left\| \left[M^{-1}F^TM^{-1}B^T\right]_j - M^{-1}B^TM_p^{-1}\left[F_p^T\right]_j \right\|_M \quad \forall j. \tag{6.10}$$

Again, by using the corresponding set of normal equations, the resulting minimiser matrix F_p is shown to be that in (6.8).

We note that splitting F_p^T into its columns and formulating the individual weighted least-squares problems for the columns is identical to using the weighted Frobenius norm $\|\cdot\|_{F,M}$ on the discrete adjoint commutator. This is because minimisation in a Frobenius norm naturally splits into independent minimisation problems for the rows or columns of the argument matrix (depending on whether it acts from the left or the right). A consequence of the above is that the resulting LSC method is independent of the choice of whether to commute with the divergence or the gradient; this is not true of the PCD method (where the operators are reversed); see Remark 5.

Before continuing, we consider what kind of underlying continuous operator is being represented by the important central term $BM^{-1}FM^{-1}B^T$ in the LSC inverse Schur complement approximation (6.9) and connect it with the PCD approach. The operator to be considered is $\mathcal{B}\mathcal{F}\mathcal{B}^T$. As with our discussions of the single-phase PCD approach, we suppose that $\boldsymbol{w}$ is constant (along with the density and viscosity) and thus, neglecting boundary conditions, obtain in the steady case that

$$\mathcal{B}\mathcal{F}\mathcal{B}^T p = \nabla \cdot (-\mu \nabla \cdot \nabla + \rho \boldsymbol{w} \cdot \nabla) \nabla p = (-\mu \nabla \cdot \nabla + \rho \boldsymbol{w} \cdot \nabla)(\nabla \cdot \nabla) p. \quad (6.11)$$

We recognise this operator as $\mathcal{F}_p\mathcal{B}\mathcal{B}^T$ from our discussion of PCD with $\mathcal{F}_p$ given as in (5.7) in the more general time-dependent case. We further note that, since in the LSC approach the same representation of the divergence B and gradient B^T is used throughout, the boundary conditions for the $\mathcal{B}\mathcal{B}^T$ terms in $\mathcal{F}_p\mathcal{B}\mathcal{B}^T(\mathcal{B}\mathcal{B}^T)^{-1}$ match and the operators cancel leaving us with

$$\mathcal{B}\mathcal{F}\mathcal{B}^T(\mathcal{B}\mathcal{B}^T)^{-1}p = \mathcal{F}_p p. \quad (6.12)$$

As such, with appropriate boundary conditions and under the assumption that we consider the single-phase Oseen problem with a constant wind $\boldsymbol{w}$, we see that the LSC and PCD preconditioning approaches are equivalent in the continuous setting.

Now let us return to the LSC approximation (6.9) and consider its practicality. The primary cost in applying the approximation is the two Laplacian solves required. Furthermore, the term $BM^{-1}B^T$ to be constructed is generally dense, thus making this term less appealing to work with than, say, the sparse Laplacian (5.10). To reduce this cost we recall that the diagonal part of a mass matrix is a good approximation to the full mass matrix. Thus we replace M by its diagonal approximation D and so require the sparse (scaled) Laplacian $BD^{-1}B^T$. This results in the standard form of the LSC inverse Schur complement

$$\widehat{S}^{-1} = \left(BD^{-1}B^T\right)^{-1} BD^{-1}FD^{-1}B^T \left(BD^{-1}B^T\right)^{-1}. \quad (6.13)$$

We note that, as with PCD, no nonsymmetric matrices are required to be inverted and here the appearance of F requires only the multiplication of matrices. However, unlike with PCD, the arguments utilised to derive the LSC preconditioner do not feature any issues of boundary conditions, primarily since all the manipulation is done after the initial discretisation. Nonetheless, it turns out that such considerations can be important in order to gain scalable behaviour of the preconditioner. We will detail how this can be achieved in the single-phase case shortly but first turn our attention to the matter of utilising stabilised finite elements where the stabilisation matrix C is non-zero.

6.1.2 Extension of the LSC approach to stabilised elements

Here we describe how the LSC methodology can be extended to handle the use of stabilised finite elements. The difficulty arises because the discrete operators B and B^T have properties that are inconsistent with those of the analogous continuous operators. We show, by way of the example of $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ elements, how to incorporate appropriate stabilisation terms in the LSC inverse Schur complement approximation (6.13). In particular, we follow the element-based stabilisation technique described in [65, 71]. We omit the analysis (which is provided in these references) to focus on the resulting modification required.

The main requirement is that both $BD^{-1}B^T$ and $BD^{-1}FD^{-1}B^T$ need stabilisation due to the inconsistency of B^T. Thus we require the form

$$\widehat{S}^{-1} = \left(BD^{-1}B^T + C_1\right)^{-1}\left(BD^{-1}FD^{-1}B^T + C_2\right)\left(BD^{-1}B^T + C_1\right)^{-1}, \qquad (6.14)$$

where C_1 and C_2 are stabilisation terms to be determined. Suitable definitions for these terms can be deduced by relating the operators they stabilise back to the Stokes operator and applying techniques analogous to those used to define C. The outcome of the analysis provided in [65, 71] is that C_1 and C_2 can be built in the same manner as C, except the need to scale appropriately with viscosity and element area.

We now suppose that we discretise using $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ elements with the appropriate stabilisation matrix featuring in the Schur complement $S = BF^{-1}B^T + C$ being given by the bilinear form in (3.50). Note that here we will maintain the dependence on viscosity inside the element integrals, as will be required in the variable coefficient case. The matrix C_1 stems from a similar form to that giving C but now divided by the element area $|T|$ and without the dependence on viscosity, namely we have

$$c_1(p_h, q_h) = \sum_{T\in\mathcal{T}_h} \frac{1}{|T|}\int_T (p_h - \Pi_0 p_h)|_T (q_h - \Pi_0 q_h)|_T, \qquad (6.15)$$

where we recall that Π_0 is the projection operator onto the space $\boldsymbol{P}_0$ given by (3.49). In a similar manner, the necessary scaling to yield C_2 is provided by the form

$$c_2(p_h, q_h) = \sum_{T\in\mathcal{T}_h} \frac{1}{|T|^2}\int_T \mu\,(p_h - \Pi_0 p_h)|_T (q_h - \Pi_0 q_h)|_T. \qquad (6.16)$$

The matrices $C_{1;i,j} = c_1(\psi_j, \psi_i)$ and $C_{2;i,j} = c_2(\psi_j, \psi_i)$ can be assembled from element contributions in the standard manner. Finally, we note that the necessary scaling within these stabilisation terms changes if a different scaling matrix D is used within the LSC approximation (6.14); this will become important later in our two-phase setting. Before moving onto the variable coefficient problem we concern ourselves with the issue of incorporating boundary conditions within the LSC approach.

6.1.3 Boundary conditions within the LSC preconditioner

As we saw for the PCD preconditioner, consideration of boundary conditions within the commutator can be crucial in order to gain a scalable approach. While the derivation of the LSC preconditioner makes no reference to boundary conditions, they are tacitly present in F_p and we know that if the wrong boundary conditions are used the performance of the preconditioner deteriorates as we refine the mesh (see also [147]). On the other hand, a relatively poor approximation (such as the pressure space mass matrix for large Reynolds number flows) which does not make use of incorrect boundary conditions can still provide scalable performance and so will ultimately be superior if the mesh is refined enough. The key point is that the minimisation problem in (6.6) does not account for the requirement of certain boundary conditions being required and seeks only to minimise the commutator, which need not align with optimising the resulting preconditioner.

When considering boundary conditions for F_p, [72] concludes that it is favourable to promote components in the commutator corresponding to the normal direction along the Dirichlet part of the boundary. This was used in Section 5.1.2 in order to choose the appropriate Robin condition for the PCD approach. For our choice of boundary conditions in (2.11) this corresponds to promoting the normal component for the section $\partial\Omega_a \cup \partial\Omega_c$ of $\partial\Omega$. In practice, this is done by de-emphasising the tangential component on this part of the boundary. The means to do so lies in our choice of norm for the weighted least-squares problem.

The Frobenius norm in (6.6) treats all entries of the commutator equally. In order to de-emphasise some components we can introduce a positive definite diagonal weight matrix W_ϵ depending on a small damping parameter $\epsilon \leq 1$. We then formulate the least-squares problem in the norm $\|\cdot\|_{F,H}$ with $H = W_\epsilon^{1/2} M^{-1} W_\epsilon^{1/2}$, namely

$$\min_{M_p^{-1}F_p} \left\| M_p^{-1} B M^{-1} F - M_p^{-1} F_p M_p^{-1} B \right\|_{F, W_\epsilon^{1/2} M^{-1} W_\epsilon^{1/2}}. \tag{6.17}$$

Following the same procedure of forming the normal equations as before, the resulting inverse Schur complement approximation is now

$$\left(BF^{-1}B^T\right)^{-1} \approx \left(BM^{-1}B^T\right)^{-1} BM^{-1}FHB^T \left(BHB^T\right)^{-1}. \tag{6.18}$$

For a more practical approach we again use the diagonal approximation to the mass matrix D and, since then the matrices in H commute to be $D_\epsilon^{-1} = W_\epsilon D^{-1}$, we obtain

$$\widehat{S}^{-1} = \left(BD^{-1}B^T\right)^{-1} BD^{-1}FD_\epsilon^{-1}B^T \left(BD_\epsilon^{-1}B^T\right)^{-1}. \tag{6.19}$$

It is interesting to note that even if the matrix F, and thus the Schur complement matrix S, are symmetric (as is true in the Stokes case) this gives a *nonsymmetric* approximation to a *symmetric* Schur complement, unless $H = M^{-1}$ and thus the requirement that $W_\epsilon = I$. Hence, we see that the application of boundary conditions results in an asymmetry of the approach. Further, results in [190] for LSC-type preconditioning for the Stokes problem demonstrate that this asymmetry is required for good performance. That is, if D is replaced by D_ϵ in (6.19), so that we regain symmetry if F is symmetric, then somewhat poorer results are obtained. We now consider details of the weighting in the Frobenius norm so as to incorporate boundary adjustments.

What remains to be determined is the diagonal weight matrix W_ϵ. Since the role of W_ϵ is to de-emphasize the tangential component of the commutator on the boundary $\partial\Omega_a \cup \partial\Omega_c$ we follow [71] to define an index set $\mathcal{J}$ corresponding to the components we wish to damp. We assume for ease that the boundary is grid-aligned, that is, always parallel to either the x-axis or the y-axis; a generalisation to non-grid-aligned domains is given in [71]. In this setting we define $\mathcal{J}$ to be such that $j \in \mathcal{J}$ if j corresponds to the index of an x-component velocity basis function on a horizontal part of the boundary $\partial\Omega_a \cup \partial\Omega_c$ *or* j corresponds to the index of a y-component velocity basis function on a vertical part of the boundary $\partial\Omega_a \cup \partial\Omega_c$. We then define W_ϵ based on $\mathcal{J}$ as the diagonal matrix with

$$W_{\epsilon;j,j} = \begin{cases} \epsilon & \text{if } j \in \mathcal{J}, \\ 1 & \text{otherwise.} \end{cases} \tag{6.20}$$

Note that ϵ should also be used to scale the appropriate stabilisation terms; in [71] a compromise is proposed between the requirement of stabilisation and the adjustments for boundary conditions given by

$$\widehat{S}^{-1} = \left(BD^{-1}B^T + C_1\right)^{-1} \left(BD^{-1}FD_\epsilon^{-1}B^T + \epsilon C_2\right) \left(BD_\epsilon^{-1}B^T + \epsilon C_1\right)^{-1}. \tag{6.21}$$

However, we remark that ϵ should not be taken too small in this case since the contributions of the stabilisation terms in the second and third set of bracketed terms in (6.21) diminish with ϵ. Regarding selection of the damping parameter, the choice $\epsilon = 0.1$ is taken in [72, 71]; this is seen to provide good behaviour in the single-phase case and results provided exhibit mesh-independent convergence. We will investigate the dependence upon ϵ in the variable coefficient case that we will shortly turn to. First, however, we consider if the LSC approach can reduce to the Cahouet–Chabard preconditioner in the Stokes case.

We saw in Section 5.1.3 that the PCD approach reduced to the appropriate Cahouet–Chabard preconditioner for the time-dependent Stokes problem. As such, it provides good performance in the Stokes limit. This also provided a helpful viewpoint in extending the PCD preconditioner to two-phase flow. For the LSC preconditioner, however, it is not clear that this provides an instructive way forward. To see this, suppose we use LSC for a single-phase Stokes problem, then $F = \mu A + \Delta t^{-1}\rho M$ where A is a discrete vector Laplacian operator. For ease of exposition, consider the LSC inverse Schur complement approximation provided in (6.9), where we have not used diagonal approximation of the mass matrices involved and no boundary adjustments are incorporated. In this case we find that

$$\begin{aligned}\widehat{S}^{-1} &= \left(BM^{-1}B^T\right)^{-1} BM^{-1}(\mu A + \Delta t^{-1}\rho M)M^{-1}B^T\left(BM^{-1}B^T\right)^{-1} \\ &= \left(BM^{-1}B^T\right)^{-1} BM^{-1}\mu A M^{-1}B^T\left(BM^{-1}B^T\right)^{-1} + \Delta t^{-1}\rho\left(BM^{-1}B^T\right)^{-1}\end{aligned} \tag{6.22}$$

and we cannot simplify the first term any further. If we consider the form of this term using the underlying differential operators we obtain $(\nabla\cdot\nabla)^{-1}\nabla\cdot(\mu\nabla\cdot\nabla)\nabla\,(\nabla\cdot\nabla)^{-1}$, which does not simplify to $\mu\mathcal{I}$ unless we can commute the central Laplacian term with either the neighbouring divergence or gradient. On the other hand, the time-stepping term in (6.22) correctly reduces to a discrete form of the time-stepping term in the Cahouet–Chabard preconditioner (c.f. (4.16)). We remark that the only choice of A which allows for the correct reduction is $A = B^T M_p^{-1} B$ but this is not a discretisation of the Laplacian operator. However, we note in passing that such a term (with a diagonalised form of mass matrix) is used in the augmented Lagrangian approach [21]. We now turn our attention to the problem of variable viscosity Stokes flow.

6.2 LSC-type preconditioning for the two-phase Stokes problem

Work on LSC-type methods for variable coefficient problems has concentrated on the variable viscosity Stokes problem, primarily in the steady case. Two-phase Stokes flow is a special case of variable viscosity flow but in some sense is distinct since we consider a viscosity field which is discontinuous, else with narrow regions of highly varying viscosity in the smoothed case, which may depend on the mesh. The main distinction in LSC-type methods aimed at problems with variable viscosity is the choice of the scaling used, namely represented in the diagonal matrix D in (6.13) or (6.19). For stabilised elements, such scaling must also be matched within C_1 and C_2 in (6.14) or (6.21).

Since it was found that the original LSC method performed relatively poorly for the variable viscosity Stokes problem, it became apparent that scaling which captures the variable coefficient nature of the problem was required. In [150] this was achieved through a rescaling of the original system matrix upon which the standard LSC approach (6.13) was applied. This effectively entails using a scaling matrix D such that $D_{i,i} = \max_j |F_{i,j}|$. In [230] it is noted that this is typically the same as taking $D = \mathrm{diag}(F)$ in practice. Hence we see that a SIMPLE-type approximation is being used for the scaling matrix. Here we will label this method $\mathrm{LSC}_{\mathrm{diag}(F)}$.

While $\mathrm{LSC}_{\mathrm{diag}(F)}$ can improve on the original approach in certain circumstances, such a choice of scaling might appear somewhat strange given our understanding of how D arises from a weighted least-squares problem to minimise a discrete version of the commutator (6.1). Indeed, a recent improvement is given in [190] which utilises a scaled mass matrix and so effectively considers a norm weighted by the viscosity coefficient.[1] In particular, analysis in [190] suggests using a mass matrix scaled by the square root of the viscosity, namely $M^{(\mu^{1/2})}$, the diagonal scaling matrix D is then given by lumping this mass matrix. We denote the resulting LSC-type method by $\mathrm{LSC}(\mu^{1/2})$ and suppose the corresponding approximate Schur complement operator is $\widehat{\mathcal{S}}(\mu^{1/2})$. We now state the main spectral equivalence theorem of [190], which is in the setting of the infinite dimensional operators rather than for the discrete operators.

Theorem 1 *Consider the stationary Stokes equations in the infinite dimensional case on a domain Ω with a sufficiently regular and bounded viscosity $\mu \in H^1(\Omega) \cap L^\infty(\Omega)$ such that $0 < \mu_{min} \leq \mu(\boldsymbol{x})$ almost everywhere in Ω. Suppose free-slip and no normal flow boundary conditions, as in* (2.11c)*, are imposed on $\partial\Omega$. Then the exact Schur complement operator $\mathcal{S}$ is equivalent to the LSC approximation $\widehat{\mathcal{S}}(\mu^{1/2})$ such that*

$$\left(\widehat{\mathcal{S}}(\mu^{1/2})q, q\right) \leq (\mathcal{S}q, q) \leq C_\mu \left(\widehat{\mathcal{S}}(\mu^{1/2})q, q\right) \text{ for all } q \in L^2(\Omega)/\mathbb{R} \cap H^1(\Omega), \quad (6.23)$$

where $(\cdot, \cdot)$ denotes the standard inner product,

$$C_\mu = \left(1 + \frac{1}{4}\|\nabla \mu\|^2_{L^\infty(\Omega)}\right)\left(C^2_{P,\mu} + 1\right) C^2_{K,\mu} \quad (6.24)$$

is a viscosity dependent constant, and $C_{P,\mu}$ and $C_{K,\mu}$ are constants stemming from weighted Poincaré–Friedrichs' and Korn's inequalities respectively; see [190].

[1] Note that, for consistency when we move to the Navier–Stokes problem, we suppose the relevant viscosity is given by μ rather than ν and that for the stationary Stokes problem the density does not feature explicitly (it may feature in the dimensionless scaling of the pressure); thus, we emphasise that in terms of the mathematical problem there is only a single variable coefficient, namely μ.

We emphasize that the constant C_μ in Theorem 1 depends on the gradient of the viscosity. In fact, Theorem 1 does not apply in the two-phase case since here the viscosity is discontinuous (not in $H^1(\Omega)$) and $\nabla\,\mu$ is unbounded. Further, the spectral equivalence holds (albeit with the constants suitably altered) when a different scaling is present in the diagonalised mass matrix D, as opposed to $\mu^{1/2}$, though this latter choice was deemed favourable in terms of the resulting constants. We further recall that this theorem is for the infinite dimensional Stokes problem with specific boundary conditions. For Dirichlet boundaries additional scaling must still be incorporated to account for the boundary conditions within the commutator, as discussed in Section 6.1.3. Thus, while Theorem 1 provides helpful guidance, it is not definitive that scaling with $\mu^{1/2}$ in D is necessarily the best choice, especially in the two-phase case, and we cannot infer that we should expect spectral equivalence or mesh-independent behaviour in the discrete case.

We now return to our primary setting of two-phase flow. Unknowing of the work in [190], we utilised a scaling of μ in the diagonalised mass matrix D in our work [27]. This choice of scaling was partly guided by our experience in the Stokes case and the aim to reduce the dependency of the LSC approach on the viscosity ratio, denoted $\widehat{\mu}$, in the absence of any boundary modifications. We shall provide evidence of this in our numerical results. Here, we denote this method by LSC(μ). Since the open choice of norm in the LSC approach leaves some scope for different methods, we will explore this issue numerically. Beforehand, however, we discuss a different approach for implementing adjustments due to boundary conditions which is given in [190].

The goal of implementing boundary adjustments remains the same, namely to damp the contribution of the commutator along certain parts of the boundary of the domain. Armed with a weighting already present in the diagonalised mass matrix, [190] modifies the weighting function in elements bordering the relevant boundary. Thus, instead of using the weighting $\mu(\boldsymbol{x})^{1/2}$, they use a weighting function

$$w_\epsilon(\boldsymbol{x}) = \begin{cases} \epsilon^{-1}\mu(\boldsymbol{x})^{1/2} & \boldsymbol{x} \in \Omega_D, \\ \mu(\boldsymbol{x})^{1/2} & \boldsymbol{x} \notin \Omega_D, \end{cases} \tag{6.25}$$

for a small parameter ϵ, where Ω_D is the union of all elements touching the Dirichlet boundary (for us $\partial\Omega_a \cup \partial\Omega_c$). The weighting w_ϵ is only used within the term D_ϵ in (6.19) as a replacement for W_ϵ, though similar weighting with a potentially different choice of ϵ is also tested within D too. However, an asymmetric approach was always seen to be preferable and retains the interpretation of changing the norm to diminish contributions from the Dirichlet boundary. We note that this strategy damps both components of the commutator (tangential and normal) on the boundary.

In essence, the two approaches considered for modification on Dirichlet boundaries are very similar, the key difference being that the original in [72] only damps tangential components while the latter in [190] damps both tangential and normal components. In terms of implementation, the former simply requires an additional diagonal matrix W_ϵ identifying the correct indices and damping appropriately while the latter requires the assembly of a second diagonalised velocity mass matrix with a different weighting function w_ϵ. To distinguish between these two approaches we suffix the label for each method with either -BC1 for the original [72] implementation and -BC2 for the latter [190] one. We now investigate both the scaling used within the LSC approach and the implementation of boundary modifications numerically.

6.2.1 Numerical experiments for Stokes flow

We consider the cavity problem described in Problem 1 in the case of two-phase Stokes flow. This test problem allows us to observe how the behaviour of LSC depends on the viscosity ratio between the two fluids, the damping parameter ϵ, and the mesh refinement. We further consider smoothing of the interface based on Theorem 1 which suggests a dependence on the thickness of the smoothing region due to the presence of the term $\|\nabla \mu\|^2_{L^\infty(\Omega)}$ in (6.24). Here, we choose to use the *inf–sup* stable pairing of square $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ elements. We detail the number of iterations required by GMRES to solve the discrete linear system to within a relative residual tolerance of 10^{-6} using the LSC preconditioner (6.19) implemented in the ideal case, that is, we suppose the scaled Laplacian sub-problems are computed using a direct method.

Based on conclusions in [190] we test with a range of choices for ϵ which decrease by a factor of two each time, namely so that $\epsilon^{-1} = 1, 2, 4, 8, \ldots, 64$. Note that $\epsilon = 1$ simply uses no boundary modifications. For the moment we detail results for the best choice of ϵ in each case, that is the minimum number of iterations observed over the range of ϵ used. We first consider the case where the viscosity is piecewise constant (thus there is no smoothing of the interface), we keep the viscosity of one fluid fixed at 1 while the viscosity of the other fluid is varied giving different viscosity ratios $\widehat{\mu}$.

Table 6.1 gives the minimum iteration counts required by GMRES over the values of ϵ used while varying the viscosity ratio $\widehat{\mu}$ and grid size h for the case of a piecewise constant viscosity field. We firstly observe that there is very little difference between the two methods to impose the boundary modifications, this suggests that only the tangential component needs to be damped but that we do not lose anything if we also damp the normal component. We can also note that, for the wide parameter regimes used, LSC(μ) requires fewer iterations than LSC($\mu^{1/2}$) except in a few cases where

Table 6.1: Preconditioned GMRES iterations when using the LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity Stokes problem (Problem 1) with piecewise constant viscosity. We vary the viscosity ratio $\widehat{\mu}$ and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments.

$\widehat{\mu}$	h	LSC(μ)-BC1	LSC(μ)-BC2	LSC($\mu^{1/2}$)-BC1	LSC($\mu^{1/2}$)-BC2
10^{-6}	$\frac{1}{8}$	7	7	17	17
	$\frac{1}{16}$	8	8	18	18
	$\frac{1}{32}$	9	9	18	18
	$\frac{1}{64}$	10	10	20	20
10^{-4}	$\frac{1}{8}$	7	7	15	15
	$\frac{1}{16}$	8	8	15	15
	$\frac{1}{32}$	9	9	15	15
	$\frac{1}{64}$	10	10	15	15
10^{-2}	$\frac{1}{8}$	7	7	9	9
	$\frac{1}{16}$	8	8	10	10
	$\frac{1}{32}$	9	9	10	10
	$\frac{1}{64}$	10	10	10	11
1	$\frac{1}{8}$	5	5	5	5
	$\frac{1}{16}$	5	5	5	5
	$\frac{1}{32}$	5	6	5	6
	$\frac{1}{64}$	5	5	5	5
10^{2}	$\frac{1}{8}$	7	7	9	9
	$\frac{1}{16}$	8	8	9	9
	$\frac{1}{32}$	9	9	9	9
	$\frac{1}{64}$	10	11	11	11
10^{4}	$\frac{1}{8}$	7	7	12	12
	$\frac{1}{16}$	8	8	11	11
	$\frac{1}{32}$	10	10	12	12
	$\frac{1}{64}$	11	11	13	13
10^{6}	$\frac{1}{8}$	7	7	12	12
	$\frac{1}{16}$	9	9	12	12
	$\frac{1}{32}$	10	10	12	12
	$\frac{1}{64}$	11	11	13	13

they require the same number of iterations (including when the viscosity ratio is 1 and thus necessarily the methods are the same, as such we subsequently ignore this case except to say that there is good mesh-independent behaviour here in the results of Table 6.1). However, this does not give the full picture as the trend with grid size h (outside of $\widehat{\mu} = 1$) suggests that LSC($\mu^{1/2}$) is more robust to mesh refinement with only a very small increase, if any, as h decreases. This contrasts with LSC(μ) which shows a more definite increase of iteration counts as the mesh is refined, albeit a small increase. On the other hand, LSC(μ) shows more robustness to the viscosity ratio $\widehat{\mu}$ with iteration counts differing by at most 1 for a wide range of values (outside of $\widehat{\mu} = 1$). Meanwhile, for LSC($\mu^{1/2}$) there is a clear dependence on $\widehat{\mu}$, especially as it approaches small values (large contrasts in viscosity in a certain direction).

Our results so far suggest that a weighting with μ within the mass matrices, as in LSC(μ), is favourable and robust to large viscosity ratio but there is some degradation as we refine the mesh. The scalability of LSC-type methods relies on the boundary adjustments being effective. We will see evidence that the behaviour of the optimal damping parameter ϵ changes in going from the single-phase case ($\widehat{\mu} = 1$) to using LSC(μ) for a two-phase problem when we consider the choice of ϵ shortly. This suggests that weighting due to the two-phase nature can interfere with the boundary adjustments. To investigate this, suppose now that the mass matrices involved in the norm defining the least-squares problem (6.17) are unscaled with respect to μ. We label these methods LSC(μ,1) and LSC($\mu^{1/2}$,1) to indicate that no scaling is used within the norm except for the boundary adjustments. For instance, LSC(μ,1) refers to the inverse Schur complement approximation (for *inf–sup* stable elements)

$$\widehat{S}^{-1} = \left(B\left(D^{(\mu)}\right)^{-1}B^T\right)^{-1} B\left(D^{(\mu)}\right)^{-1} F D_\epsilon^{-1} B^T \left(B D_\epsilon^{-1} B^T\right)^{-1}, \tag{6.26}$$

where $D^{(\mu)}$ is the diagonal approximation of the viscosity scaled mass matrix $M^{(\mu)}$ and D_ϵ remains the same as in (6.19).

Table 6.2 provides results for these new methods and compares with results in Table 6.1 but now omitting the $\widehat{\mu} = 1$ case (since the methods are identical here) and including a further grid refinement level. We see a clear improvement in going from LSC(μ) to LSC(μ,1) with the latter method not only showing excellent independence of the viscosity ratio $\widehat{\mu}$ but now scalability as we refine the mesh. For LSC($\mu^{1/2}$,1), however, iteration counts are predominantly worse than those for LSC($\mu^{1/2}$), though arguably the mesh scalability is slightly improved. Furthermore, LSC($\mu^{1/2}$,1) retains a clear dependence on the viscosity ratio. As such, we see that LSC(μ,1) is the superior choice. Again, both methods to apply boundary adjustments perform similarly.

Table 6.2: Preconditioned GMRES iterations when using the LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity Stokes problem (Problem 1) with piecewise constant viscosity. We vary the viscosity ratio $\widehat{\mu}$ and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments.

		LSC(μ)		LSC(μ,1)		LSC($\mu^{1/2}$)		LSC($\mu^{1/2}$,1)	
$\widehat{\mu}$	h	-BC1	-BC2	-BC1	-BC2	-BC1	-BC2	-BC1	-BC2
10^{-6}	$\frac{1}{8}$	7	7	7	7	17	17	38	40
	$\frac{1}{16}$	8	8	8	8	18	18	62	62
	$\frac{1}{32}$	9	9	8	8	18	18	74	74
	$\frac{1}{64}$	10	10	8	8	20	20	70	71
	$\frac{1}{128}$	11	11	7	7	22	22	63	63
10^{-4}	$\frac{1}{8}$	7	7	7	7	15	15	27	27
	$\frac{1}{16}$	8	8	8	8	15	15	33	33
	$\frac{1}{32}$	9	9	8	8	15	15	32	32
	$\frac{1}{64}$	10	10	8	8	15	15	30	30
	$\frac{1}{128}$	11	11	7	7	16	16	25	25
10^{-2}	$\frac{1}{8}$	7	7	7	7	9	9	12	12
	$\frac{1}{16}$	8	8	8	7	10	10	12	12
	$\frac{1}{32}$	9	9	8	8	10	10	11	11
	$\frac{1}{64}$	10	10	7	7	10	11	11	11
	$\frac{1}{128}$	11	11	7	7	11	11	10	10
10^{2}	$\frac{1}{8}$	7	7	7	7	9	9	12	12
	$\frac{1}{16}$	8	8	7	7	9	9	11	11
	$\frac{1}{32}$	9	9	7	7	9	9	9	9
	$\frac{1}{64}$	10	11	6	6	11	11	8	8
	$\frac{1}{128}$	10	10	6	6	11	11	7	7
10^{4}	$\frac{1}{8}$	7	7	7	7	12	12	30	30
	$\frac{1}{16}$	8	8	7	7	11	11	30	30
	$\frac{1}{32}$	10	10	7	7	12	12	28	28
	$\frac{1}{64}$	11	11	6	6	13	13	23	23
	$\frac{1}{128}$	11	11	6	6	12	12	18	18
10^{6}	$\frac{1}{8}$	7	7	7	7	12	12	44	44
	$\frac{1}{16}$	9	9	7	7	12	12	56	56
	$\frac{1}{32}$	10	10	7	7	12	12	60	61
	$\frac{1}{64}$	11	11	6	6	13	13	55	56
	$\frac{1}{128}$	11	11	6	6	13	13	48	48

The results of Table 6.2 are for the case of a piecewise constant viscosity field. We now consider a smoothed viscosity field where we transition from one fluid to another over an interface of width 2δ. We suppose that this width expands over three elements, setting $2\delta = 3h$ where h is the side length of the square elements used, and that we transition using the regularised Heaviside function (2.25); see Remark 3. In Table 6.3 we see that this smoothing of the interface can have a dramatic effect on the iteration counts required by the LSC-type preconditioners, especially when the contrast in viscosity between the two fluids is large. In such cases there is now a more prominent mesh dependence. This might have be anticipated by considering the bounds of Theorem 1 which depend on maximum value of the gradient of viscosity: as we refine the mesh we narrow the transition region and so steepen the gradient of viscosity here, thus we might expect more iterations to be required due to the worsening bound. Nonetheless, it is interesting to note this seems not to have been a prominent issue with the piecewise constant case where the gradient of viscosity is effectively infinite in the continuous case, else very large if sampling on a discrete level.

The main casualty in moving from a piecewise constant viscosity to a smoothed viscosity is the LSC(μ,1) and LSC($\mu^{1/2}$,1) methods for the most extreme viscosity contrasts. In particular, LSC(μ,1) now shows some dependence on the viscosity ratio $\widehat{\mu}$. While for $10^{-4} \leq \widehat{\mu} \leq 10^4$ performance is still scalable and by the finest mesh the iteration counts are at most one more than the piecewise constant case, outside of this range of $\widehat{\mu}$ the results significantly deteriorate and depend poorly on the mesh. This suggests that, when the interface becomes narrow and the viscosity gradient becomes steep enough, the methods which utilise a least-squares norm independent of the viscosity can fail and lack robustness. On the other hand, LSC(μ) suffers only a small degradation moving to the smoothed interface case and remains reasonably robust. Despite this, so long as the viscosity contrast is not too extreme, the most favourable method remains the LSC(μ,1) approach. For extreme viscosity contrasts, however, LSC(μ) is by far the most appropriate choice and requires relatively few iterations for convergence even if there is a small dependence on the mesh.

A similar picture of degradation is seen for the methods employing the $\mu^{1/2}$ scaling suggested in [190], though results are generally poorer. We note that LSC($\mu^{1/2}$,1) also provides scalable results for $10^{-4} \leq \widehat{\mu} \leq 10^4$ (while LSC($\mu^{1/2}$) does not) but again begins to fail for the most extreme viscosity ratios. In the case of $\widehat{\mu} = 10^{-6}$ on the finest mesh, convergence of GMRES had not occurred after 500 iterations and so the solver was deemed to fail, denoted by $\times$ in Table 6.3.

Table 6.3: Preconditioned GMRES iterations when using the LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity Stokes problem (Problem 1) with a smoothed viscosity field. We vary the viscosity ratio $\widehat{\mu}$ and grid size h. Note that $\times$ indicates a failure and iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments.

		LSC(μ)		LSC(μ,1)		LSC($\mu^{1/2}$)		LSC($\mu^{1/2}$,1)	
$\widehat{\mu}$	h	-BC1	-BC2	-BC1	-BC2	-BC1	-BC2	-BC1	-BC2
10^{-6}	$\frac{1}{8}$	14	14	31	31	21	21	36	36
	$\frac{1}{16}$	15	15	91	91	35	35	90	90
	$\frac{1}{32}$	15	15	219	216	43	43	186	187
	$\frac{1}{64}$	15	16	357	361	49	49	370	372
	$\frac{1}{128}$	18	18	412	411	56	56	$\times$	$\times$
10^{-4}	$\frac{1}{8}$	10	10	12	13	13	13	16	16
	$\frac{1}{16}$	12	12	12	12	15	15	22	22
	$\frac{1}{32}$	13	13	11	11	17	17	25	25
	$\frac{1}{64}$	14	14	10	10	21	21	24	24
	$\frac{1}{128}$	15	15	8	8	24	24	23	23
10^{-2}	$\frac{1}{8}$	8	8	8	8	7	7	9	9
	$\frac{1}{16}$	10	10	7	7	9	8	9	9
	$\frac{1}{32}$	11	11	7	7	10	10	10	10
	$\frac{1}{64}$	12	12	7	7	11	11	10	10
	$\frac{1}{128}$	12	12	6	6	11	11	9	9
10^{2}	$\frac{1}{8}$	8	8	7	7	7	7	8	8
	$\frac{1}{16}$	9	9	7	7	8	8	8	8
	$\frac{1}{32}$	9	9	6	6	9	9	8	8
	$\frac{1}{64}$	11	11	6	6	10	10	8	8
	$\frac{1}{128}$	11	11	6	6	11	11	7	7
10^{4}	$\frac{1}{8}$	9	9	12	12	12	12	17	17
	$\frac{1}{16}$	9	9	10	10	14	14	19	19
	$\frac{1}{32}$	10	10	8	8	15	15	19	19
	$\frac{1}{64}$	11	11	7	7	17	17	18	18
	$\frac{1}{128}$	11	11	7	7	17	17	16	16
10^{6}	$\frac{1}{8}$	12	12	85	85	26	26	81	81
	$\frac{1}{16}$	12	12	146	146	29	29	113	113
	$\frac{1}{32}$	11	11	169	169	31	31	163	163
	$\frac{1}{64}$	11	11	263	258	33	33	196	196
	$\frac{1}{128}$	11	11	321	320	34	34	164	164

To verify that the worsening behaviour observed in Table 6.3 upon refining the mesh is due to the narrowing of the interface we consider fixing δ independently of the mesh, given by the choice previously made when $h = 1/16$; we denote this the "fixed smoothed viscosity" case. Results for this case are displayed in Table 6.4 where we see that, once the mesh becomes fine enough (past $h = 1/16$) the iteration counts drop to be at a level much closer to those in Table 6.2 and iteration counts typically decrease slightly upon further refinement. In particular, this is notable for both the LSC(μ,1) and LSC($\mu^{1/2}$,1) when we have a large viscosity contrast since now in the extreme cases we regain scalability. Furthermore, LSC(μ,1) is now preferable to LSC(μ) for all viscosity ratios used. For the approaches utilising a scaling of $\mu^{1/2}$, we note that there is still a considerable dependence on the viscosity ratio $\widehat{\mu}$ and so these methods remain less favourable.

An interesting feature we see in Table 6.4 is that for the more extreme viscosity ratios considered there is a large jump in iteration counts seen in the LSC(μ,1) and LSC($\mu^{1/2}$,1) methods for the grid size $h = 1/16$ before reducing back to a lower level at the next refinement. This grid is precisely the one used in order to give the fixed smoothed viscosity field with $2\delta = 3h$. Thus, we see that if the interface transition region is small enough compared to the grid (as for $h = 1/8$ or the piecewise constant case) or the transition region is wide enough that the grid can well resolve this region we obtain reasonable behaviour. On the other hand, between these two cases (here $h = 1/16$) much poorer performance is seen. We remark this is only for the most extreme viscosity ratios used.

Overall, these observations imply that the lack of robustness found for LSC(μ,1) (as well as LSC($\mu^{1/2}$,1)) stems not from simply having a large viscosity contrast but from the sharpness (gradient) of viscosity coefficient over the interface relative to the element size used. That is, the mesh must be sufficiently refined to capture the transition region of the viscosity and more refinement is required for larger viscosity contrast. Thus, our results suggest that the LSC(μ,1) can be used to obtain scalable behaviour providing that the smoothing of the interface is done over a large enough number of elements for a particular viscosity ratio used. We also note here that results (not detailed here) for a smooth Gaussian viscosity field show good mesh-independent behaviour for all LSC methods considered with similar iterations counts when varying the viscosity ratio up to three orders of magnitude either side of unity; as is consistent with [190]. This suggests that each weighting of LSC used can show mesh-independent behaviour (with the correct damping parameter ϵ) so long as the viscosity field is smooth enough relative to the mesh.

Table 6.4: Preconditioned GMRES iterations when using the LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity Stokes problem (Problem 1) with a fixed smoothed viscosity field. We vary the viscosity ratio $\widehat{\mu}$ and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments.

		LSC(μ)		LSC(μ,1)		LSC($\mu^{1/2}$)		LSC($\mu^{1/2}$,1)	
$\widehat{\mu}$	h	-BC1	-BC2	-BC1	-BC2	-BC1	-BC2	-BC1	-BC2
10^{-6}	$\frac{1}{8}$	12	12	28	28	35	35	56	56
	$\frac{1}{16}$	15	15	91	91	35	35	89	89
	$\frac{1}{32}$	12	12	11	11	37	37	50	50
	$\frac{1}{64}$	12	12	10	10	33	33	45	45
	$\frac{1}{128}$	14	14	9	9	26	26	42	42
10^{-4}	$\frac{1}{8}$	12	12	27	27	20	21	29	29
	$\frac{1}{16}$	12	12	12	12	15	15	22	22
	$\frac{1}{32}$	12	12	10	10	15	15	21	21
	$\frac{1}{64}$	11	11	8	8	13	13	19	19
	$\frac{1}{128}$	11	11	7	7	12	12	17	17
10^{-2}	$\frac{1}{8}$	8	8	8	8	8	8	10	10
	$\frac{1}{16}$	10	10	7	7	9	8	9	9
	$\frac{1}{32}$	9	9	7	7	8	8	9	9
	$\frac{1}{64}$	8	8	7	6	8	8	9	9
	$\frac{1}{128}$	7	7	6	6	7	8	8	8
10^{2}	$\frac{1}{8}$	7	7	8	8	8	8	9	8
	$\frac{1}{16}$	9	9	7	7	8	8	8	8
	$\frac{1}{32}$	7	7	6	6	7	7	8	8
	$\frac{1}{64}$	7	7	6	6	7	7	7	7
	$\frac{1}{128}$	7	7	6	6	7	7	6	6
10^{4}	$\frac{1}{8}$	9	9	35	35	20	20	51	51
	$\frac{1}{16}$	9	9	10	10	14	14	19	19
	$\frac{1}{32}$	10	10	8	8	14	14	17	17
	$\frac{1}{64}$	10	10	7	7	12	12	15	15
	$\frac{1}{128}$	10	10	6	6	11	11	13	13
10^{6}	$\frac{1}{8}$	10	10	34	34	36	36	66	67
	$\frac{1}{16}$	12	12	146	146	29	29	113	114
	$\frac{1}{32}$	10	10	20	20	27	27	73	73
	$\frac{1}{64}$	11	11	10	10	22	22	35	35
	$\frac{1}{128}$	12	12	6	6	16	16	28	28

A final issue we have yet to discuss is that of the choice of damping parameter ϵ. Results in [190] for smooth viscosity fields suggest that, as the grid size h is halved, the value of ϵ should also be halved. This strategy is exemplified in our results for the case of $\widehat{\mu} = 1$ (which will be omitted from the following tables) where, to be explicit, the optimal value(s) of ϵ^{-1} are given by $\{4, 8\}$, $\{8\}$, $\{16\}$, $\{32\}$, and $\{32, 64\}$ for $h = 1/8$ refining to $h = 1/128$ respectively for our -BC1 option.

To investigate this for our two-phase problem, Tables 6.5 and 6.6 shows the value(s) of ϵ^{-1} which give the minimum iteration counts in each case for the range of interface choices used. In these results we restrict ourselves to the original boundary adjustment technique of [72]; we remark that a broadly similar picture is found for the other boundary adjustment technique. Table 6.5 gives results for the LSC(μ) and LSC($\mu^{1/2}$) methods where we previously observed a difficulty in obtaining mesh-independent results due the additional viscosity scaling in the least-squares norm. We immediately see that the optimal ϵ^{-1} primarily does not change with the grid size h in contrast to the single-phase case. This evidences a detrimental interplay between such viscosity scaling and the scaling due to ϵ. For both methods, choosing ϵ^{-1} fairly close to 1 appears best, in particular $\epsilon^{-1} = 2$ is almost always optimal, but, as we saw previously, typically does not yield mesh-independence.

Table 6.5 gives results for the LSC(μ,1) and LSC($\mu^{1/2}$,1) methods. For LSC($\mu^{1/2}$,1) we see similar results to the previous methods with a small ϵ^{-1} being best. However, for LSC(μ,1) we regain optimal values of ϵ^{-1} which roughly double as we halve the grid size h. Further, for the piecewise constant case we see a much wider range of optimal ϵ, indicating that performance is not overly sensitive to ϵ. Outside of the breakdown for extreme viscosity contrasts in the smoothed case, the doubling pattern is observed for the other interface choices. This suggests that the LSC(μ,1) best extends the original LSC approach to our two-phase Stokes problem and further links with the improved scalability and performance we observed in the iteration counts (in almost all cases). Finally, we note that, for all methods, typically if the damping parameter changes by a factor of two from an optimal value the iteration counts are fairly similar but if it moves further away then iteration counts can be notably worse.

In conclusion, we reiterate that for two-phase problems using a μ scaling within LSC is preferable to the $\mu^{1/2}$ scaling suggested in [190]. Further, mesh-independence is only gained when using a least-square norm that does not depend on the viscosity, so long as the mesh is refined enough to capture any smoothing of the interface. In particular, LSC(μ,1) performs best unless the mesh refinement is not sufficient to capture the sharpness of the interface, in which case LSC(μ) is favourable.

Table 6.5: Optimal inverse damping parameter(s) ϵ^{-1} from $\epsilon^{-1} \in \{1, 2, 4, 8, \ldots, 64\}$ for varying viscosity ratios $\widehat{\mu}$, grid size h and choice of interface smoothing: PC denotes piecewise constant (no smoothing), S denotes a smoothed interface with $2\delta = 3h$, and FS denotes a smoothed interface of fixed width given by $2\delta = 3/16$.

		LSC(μ)-BC1			LSC($\mu^{1/2}$)-BC1		
$\widehat{\mu}$	h	PC	S	FS	PC	S	FS
10^{-6}	$\frac{1}{8}$	2	1, 2	1, 2	1, 2, 4	1	1
	$\frac{1}{16}$	2	1, 2	1, 2	1, 2, 4	1	1
	$\frac{1}{32}$	2	1, 2	2, 4	1	1	1, 2, 4
	$\frac{1}{64}$	2	4	2, 4	1, 2, 8	1	1
	$\frac{1}{128}$	2, 4	1, 2	4, 8	2, 4, 8	1, 2	1, 2
10^{-4}	$\frac{1}{8}$	2	1, 2	1, 2, 4	1, 2, 4	1, 2	2
	$\frac{1}{16}$	2	1, 2, 4	1, 2, 4	1, 2, 4	1, 2, 4	1, 2, 4
	$\frac{1}{32}$	2	2, 4	1, 2, 4	1, 2	2, 4	1, 2, 4
	$\frac{1}{64}$	2	1, 2, 4	4	4	1, 2, 4	2, 4
	$\frac{1}{128}$	2, 4	1, 2, 4	2, 4, 8	4	1, 2	4, 8
10^{-2}	$\frac{1}{8}$	2	2	2	1, 2	2, 4	1, 2
	$\frac{1}{16}$	2	1, 2, 4	1, 2, 4	1, 2, 4	1, 2, 4, 8	1, 2, 4, 8
	$\frac{1}{32}$	2	1, 2, 4	1, 2, 4, 8	2	1, 2, 4, 8	4
	$\frac{1}{64}$	2	1, 2, 4	4	2	2, 4, 8	4, 8
	$\frac{1}{128}$	2, 4	2	4	2, 4	2, 4, 8	16
10^{2}	$\frac{1}{8}$	1, 2, 4	1, 2, 4	1, 2	1, 2, 4	1, 2	1, 2
	$\frac{1}{16}$	1, 2, 4	1, 2, 4	1, 2, 4	1, 2, 4	1, 2, 4	1, 2, 4
	$\frac{1}{32}$	1	1	2	1, 2	1, 2, 4	2
	$\frac{1}{64}$	2	1, 2	4	1, 2	2	2, 4
	$\frac{1}{128}$	2	1, 2	4, 8, 16	1, 2	1, 2, 4	2, 4, 8, 16
10^{4}	$\frac{1}{8}$	1, 2, 4	1, 2, 4	1, 2	1, 2, 4	1	1, 2
	$\frac{1}{16}$	1, 2	1, 2	1, 2	1	1, 2	1, 2
	$\frac{1}{32}$	1, 2, 4	1, 2	1, 2	1, 2	2	1, 2
	$\frac{1}{64}$	1, 2, 4	1, 2, 4	2	1, 2, 4	1, 2, 4	1, 2, 4
	$\frac{1}{128}$	1, 2, 4	1, 2	1, 2, 4	1	1	1, 2
10^{6}	$\frac{1}{8}$	1, 2, 4	1, 2	1, 2	1, 2	1, 2	1
	$\frac{1}{16}$	1, 2, 4	1, 2	1, 2	1, 2, 4	2	2
	$\frac{1}{32}$	1, 2, 4	1	1, 2	2	1, 2	1, 2
	$\frac{1}{64}$	1, 2, 4	1, 2	1, 2, 4	1, 2, 4	1	1
	$\frac{1}{128}$	1, 2, 4	1, 2	1, 2, 4	1, 2	1, 2	1

Table 6.6: Optimal inverse damping parameter(s) ϵ^{-1} from $\epsilon^{-1} \in \{1, 2, 4, 8, \ldots, 64\}$ for varying viscosity ratios $\widehat{\mu}$, grid size h and choice of interface smoothing: PC denotes piecewise constant (no smoothing), S denotes a smoothed interface with $2\delta = 3h$, and FS denotes a smoothed interface of fixed width given by $2\delta = 3/16$.

		LSC(μ,1)-BC1			LSC($\mu^{1/2}$,1)-BC1		
$\widehat{\mu}$	h	PC	S	FS	PC	S	FS
10^{-6}	$\frac{1}{8}$	4	1	1, 2	4	1, ..., 8	1
	$\frac{1}{16}$	4, ..., 32	8	8	4	1	1
	$\frac{1}{32}$	4, ..., 64	4	8	4	2	1
	$\frac{1}{64}$	8, ..., 64	16	16, 32	4	2	2
	$\frac{1}{128}$	16, 32, 64	8	16, 32, 64	4	×	4
10^{-4}	$\frac{1}{8}$	4	4	2	2, 4	2	1
	$\frac{1}{16}$	4, ..., 32	2, 4	2, 4	1	1, 2, 4	1, 2, 4
	$\frac{1}{32}$	4, ..., 64	8	8, 16, 32	2	1, 2	1, 2, 4
	$\frac{1}{64}$	8, ..., 64	8, 16	8, 16, 32	1, 2, 4	1, 2	2
	$\frac{1}{128}$	16, 32, 64	16	32	1, 2	2, 4	4
10^{-2}	$\frac{1}{8}$	2, 4	2, 4, 8	2, 4, 8	2	1, ..., 8	1, ..., 8
	$\frac{1}{16}$	2, ..., 64	4, 8, 16	4, 8, 16	1, 2, 4	2	2
	$\frac{1}{32}$	4, ..., 64	8, 16	8, 16, 32	2	2, 4, 8	2, 4
	$\frac{1}{64}$	16	8, ..., 64	8, ..., 64	2, 4	2, 4, 8	2, 4, 8
	$\frac{1}{128}$	16, 32, 64	16	16, 32, 64	2, 4, 8	4, 8	4, 8, 16
10^{2}	$\frac{1}{8}$	1, 2, 4	2	1, 2, 4	1, 2, 4	1, 2, 4	1, 2, 4
	$\frac{1}{16}$	2, 4	2, 4	2, 4	1, 2, 4	1, 2, 4	1, 2, 4
	$\frac{1}{32}$	2, 4, 8	4	2, 4	1	1, 2, 4	1, 2, 4
	$\frac{1}{64}$	4, 8	4, 8	4, 8	2	2, 4, 8	2, 4
	$\frac{1}{128}$	8, ..., 64	8, ..., 64	4, ..., 32	4	4, 8	4
10^{4}	$\frac{1}{8}$	1, 2	1	1	1, 2	1	4
	$\frac{1}{16}$	2, 4	1, 2	1, 2	1	1, 2	1, 2
	$\frac{1}{32}$	2, 4, 8	2	2, 4	1, 2	1	1, 2
	$\frac{1}{64}$	4, 8	4	2, ..., 16	1	1, 2	1, 2, 4
	$\frac{1}{128}$	8, ..., 64	4, 8, 16	4, 8	1	1, 2, 4	1, 2, 4
10^{6}	$\frac{1}{8}$	1, 2	2	1	2	1	2
	$\frac{1}{16}$	2, 4	1, 64	1	2	2	2
	$\frac{1}{32}$	2, 4, 8	1	2, 4, 8	4	2	1
	$\frac{1}{64}$	4, 8	16, 64	8, 16, 32	4	2	1
	$\frac{1}{128}$	8, ..., 64	2	4, 8	4	1	1

6.3 LSC-type preconditioning for the two-phase Navier–Stokes problem

We now return to the case of Navier–Stokes flow. Note that now we have two variable coefficients, the density and viscosity, which can be incorporated in scaling of the LSC methodology. Before exploring this further, we first return to the original commutator idea generating the LSC approach. The commutator used in the single-phase case is given in (6.1), namely given by $\mathcal{B}\mathcal{F} - \mathcal{F}_p\mathcal{B}$. However, from our discussion of the PCD approach in Section 5.3.3 we know that a more fitting commutator to begin with in the two-phase case is the bi-weighted commutator

$$\mathcal{B}\rho^{-1}\mathcal{F} \approx \mathcal{F}_p\mu\mathcal{B}. \tag{6.27}$$

Discretising this commutator and applying the otherwise standard LSC approach we arrive at the weighted least-squares problem

$$\min_{M_p^{-1}F_p} \left\| M_p^{-1}B\left(M^{(\rho)}\right)^{-1}F - M_p^{-1}F_p\left(M_p^{(1/\mu)}\right)^{-1}B \right\|_{F,H}, \tag{6.28}$$

where we leave the weighting H in the Frobenius norm unspecified for now. Solving this least-squares minimisation problem yields

$$F_p = B\left(M^{(\rho)}\right)^{-1}FHB^T\left(BHB^T\right)^{-1}M_p^{(1/\mu)}. \tag{6.29}$$

Meanwhile, approximating the (weighted) discrete commutator to be zero we obtain

$$\left(BF^{-1}B^T\right)^{-1} \approx \left(B\left(M^{(\rho)}\right)^{-1}B^T\right)^{-1}F_p\left(M_p^{(1/\mu)}\right)^{-1}. \tag{6.30}$$

Combining (6.29) and (6.30) we derive the inverse Schur complement approximation

$$\left(BF^{-1}B^T\right)^{-1} \approx \left(B\left(M^{(\rho)}\right)^{-1}B^T\right)^{-1}B\left(M^{(\rho)}\right)^{-1}FHB^T\left(BHB^T\right)^{-1}. \tag{6.31}$$

We note that this does not depend on the choice of weight between $\mathcal{F}_p$ and $\mathcal{B}$ on the right-hand side of (6.27) since this always cancels. Thus, outside of the choice of norm, we see that utilising the bi-weighted commutator only yields a scaling with density in the Schur complement approximation. Furthermore, looking back to the LSC-type methods proposed in Section 6.2 for the Stokes problem, we see that they implicitly make use of a viscosity weighted commutator which incorporates $\mathcal{B}\mu^{-1/2}\mathcal{F}$ or $\mathcal{B}\mu^{-1}\mathcal{F}$ for LSC($\mu^{1/2}$) or LSC(μ) respectively. This suggests that the improved dependence on the viscosity ratio seen in LSC(μ) may be due to the weight μ^{-1} effectively trying to cancel the dependence on viscosity in the commutator since here we have

$$\mathcal{B}\mu^{-1}\mathcal{F} = \nabla\cdot\left(-\mu^{-1}\nabla\cdot\left(2\mu\,\boldsymbol{D}\boldsymbol{u}\right)\right) \approx \nabla\cdot\left(-\nabla\cdot\left(2\,\boldsymbol{D}\boldsymbol{u}\right)\right). \tag{6.32}$$

As before, in order obtain a practical method, diagonalisation of the velocity mass matrices and in H is required. As such, we let $D^{(\rho)}$ be the diagonalised form of $M^{(\rho)}$ and further, for clarity of expression, define its inverse to be $V^{(\rho)}$. We suppose that the weighting H in the Frobenius norm derives from a weighted mass matrix $M^{(\xi)}$ for some ξ so that, incorporating boundary conditions as discussed previously, H is replaced by the inverse of $D_\epsilon^{(\xi)}$ which we define as $V_\epsilon^{(\xi)}$. Hence the practical LSC-type inverse Schur complement approximation is given by

$$\widehat{S}^{-1} = \left(BV^{(\rho)}B^T\right)^{-1} BV^{(\rho)}FV_\epsilon^{(\xi)}B^T \left(BV_\epsilon^{(\xi)}B^T\right)^{-1}. \tag{6.33}$$

While the choice of the bi-weighted commutator (6.27) was derived to be suitable for the PCD approach, no complications arise for considering other choices of weights for the LSC methodology. Further, (6.33) does not incorporate, for instance, the LSC(μ) method introduced previously in Section 6.2. Thus, to give a more general LSC-type methodology, we suppose the generic commutator (5.66) is our starting point where we now know that the weight π is the only weight which will enter the resulting approximation. Now, when considering the stabilised formulation required for $C = 0$, the additional scaling that is incorporated in the various terms must also be taken into account in C_1 and C_2 so as to ensure these terms are commensurate with the terms they stabilise. Thus, the most general form for a two-phase LSC approach that we consider uses the inverse Schur complement approximation $\widehat{S}^{-1}$ given by

$$\left(BV^{(1/\pi)}B^T + C_1^{(\pi)}\right)^{-1}\left(BV^{(1/\pi)}FV_\epsilon^{(\xi)}B^T + \epsilon C_2^{(\pi/\xi)}\right)\left(BV_\epsilon^{(\xi)}B^T + \epsilon C_1^{(1/\xi)}\right)^{-1}, \tag{6.34}$$

where the scaled stabilisation terms (with generic weight η) derive from

$$c_1^{(\eta)}(p_h, q_h) = \sum_{T\in\mathcal{T}_h} \frac{1}{|T|}\int_T \eta\,(p_h - \Pi_0 p_h)|_T(q_h - \Pi_0 q_h)|_T, \tag{6.35}$$

$$c_2^{(\eta)}(p_h, q_h) = \sum_{T\in\mathcal{T}_h} \frac{1}{|T|^2}\int_T \eta\,\mu\,(p_h - \Pi_0 p_h)|_T(q_h - \Pi_0 q_h)|_T, \tag{6.36}$$

in the case of $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ elements. Consistently with Section 6.2, we label this approach as LSC(π^{-1}, ξ) for specific values of π^{-1} and ξ, which are the two scalings used in the (diagonalised) mass matrices. When $\pi^{-1} = \xi$, so that the scaling used is the same, we collapse this label into LSC(ξ) as with the previously defined LSC-type methods such as LSC(μ). Note that outside of the case $\pi^{-1} = \xi$, two separate scalings of the C_1 term are required when considering stabilised elements. Finally, we emphasise that the scaling shown for each V term is the scaling used to construct the mass matrix before it is approximated by a diagonal matrix and inverted.

While there are many choices of π and ξ, the new methods we consider in this section all use $\pi = \rho^{-1}$ based on the bi-weighted commutator (6.27). In choosing ξ we may wish to balance the scaling in the two Laplacian terms and so select $\xi = \rho$ giving LSC(ρ). On the other hand, we may wish to omit any weighting in the norm and select $\xi = 1$, this ensures when taking into account boundary adjustments that no additional weighting is present and so, based on our Stokes results, we might anticipate improved scalability. However, for the Navier–Stokes problem we now have two variable coefficients and so choosing $\xi = 1$ limits our ability to use these two as separate weights. Instead, we might consider using a μ based weight for ξ to incorporate both ρ and μ. Alternatively, the norms used for the Stokes case considered in Section 6.2 may be used where $\pi^{-1} = \mu$ or $\pi^{-1} = \mu^{1/2}$. Thus, along with the methods LSC(μ), LSC($\mu^{1/2}$), LSC($\mu, 1$) and LSC($\mu^{1/2}, 1$) previously introduced, we consider LSC(ρ), LSC($\rho, 1$), LSC(ρ, μ), and LSC($\rho, \mu^{1/2}$).

6.3.1 Numerical experiments for Navier–Stokes flow

As with the Stokes case, we focus on the steady cavity problem described in Problem 1, now in the case of Navier–Stokes flow, and consider air–water flow with varying Reynolds numbers. We use Newton linearisation and tabulate the average number of GMRES iterations required over the nonlinear iteration, which is terminated once the nonlinear residual drops below a relative tolerance of 10^{-10}. We omit the initial Stokes solve that provides the initial guess and note that the resulting average iteration counts have been rounded; the relative linear tolerance used within GMRES is again 10^{-6}. For the moment, we continue to discuss the ideal implementation of these LSC-type methods and utilise $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ finite elements. We use the original boundary adjustment technique in [72] and note that nearly identical results apply for the adjustment technique described in [190].

Tables 6.7 to 6.9 detail performance of the LSC-type methods considered for the piecewise constant coefficient, smoothed interface, and fixed smoothed interface cases respectively. Iteration counts are given for the best choice of damping parameter ϵ used in each instance and the minimum of the average iteration counts for the different methods is highlighted in bold to more easily discern the most efficient choice(s) of scaling. We first note that we omit results for LSC($\mu^{1/2}, 1$), since it never beats LSC($\mu, 1$) as might be expected from our Stokes results, as well as results for LSC(ρ), which never beats LSC(μ). Overall, we see that the most efficient method is LSC($\mu, 1$) in most cases except for large Reynolds numbers where LSC($\rho, \mu^{1/2}$) typically requires fewer iterations, especially in the fixed smoothed interface case. Nonetheless, if we

Table 6.7: Average preconditioned GMRES iterations after Newton linearisation when using the LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) using piecewise constant coefficients with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments. Iteration counts in bold show which method is most efficient.

Re	h	LSC(μ)	LSC($\mu^{1/2}$)	LSC($\mu, 1$)	LSC($\rho, 1$)	LSC(ρ, μ)	LSC($\rho, \mu^{1/2}$)
10	$\frac{1}{8}$	12	12	**10**	18	12	12
	$\frac{1}{16}$	14	14	**11**	19	14	14
	$\frac{1}{32}$	17	17	**13**	19	17	17
	$\frac{1}{64}$	21	20	**14**	19	21	21
$10^{1.5}$	$\frac{1}{8}$	14	14	**13**	21	14	14
	$\frac{1}{16}$	16	16	**14**	22	17	16
	$\frac{1}{32}$	18	18	**15**	22	19	19
	$\frac{1}{64}$	22	22	**17**	23	22	22
100	$\frac{1}{8}$	18	20	**17**	26	18	18
	$\frac{1}{16}$	21	22	**18**	27	21	21
	$\frac{1}{32}$	23	24	**20**	28	24	23
	$\frac{1}{64}$	28	29	**21**	29	28	27
$10^{2.5}$	$\frac{1}{8}$	**25**	33	26	33	26	**25**
	$\frac{1}{16}$	28	32	**27**	35	29	**27**
	$\frac{1}{32}$	32	34	**28**	36	33	31
	$\frac{1}{64}$	38	39	**30**	37	38	36
1000	$\frac{1}{8}$	**49**	69	59	58	**49**	**49**
	$\frac{1}{16}$	**50**	69	57	58	**50**	**50**
	$\frac{1}{32}$	46	55	48	57	46	**44**
	$\frac{1}{64}$	52	58	**49**	58	53	**49**

look at mesh-dependence, LSC($\rho, 1$) proves slightly more scalable than LSC($\mu, 1$) even though it yields iteration counts that are somewhat higher. Despite this, LSC($\mu, 1$) only exhibits a very mild increase in iteration counts for the piecewise constant and smoothed interface cases while LSC($\rho, \mu^{1/2}$) displays a slow but clear degradation here. In the fixed smoothed interface case all methods show effectively mesh-independent behaviour past the $h = 1/16$ grid, consistent with our Stokes results for comparative contrasts in the variable coefficients and [190].

Table 6.8: Average preconditioned GMRES iterations after Newton linearisation when using the LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) using a smoothed interface with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments. Iteration counts in bold show which method is most efficient.

Re	h	LSC(μ)	LSC($\mu^{1/2}$)	LSC($\mu, 1$)	LSC($\rho, 1$)	LSC(ρ, μ)	LSC($\rho, \mu^{1/2}$)
10	$\frac{1}{8}$	12	**9**	10	12	13	11
	$\frac{1}{16}$	15	12	**11**	13	17	13
	$\frac{1}{32}$	19	15	**11**	13	21	16
	$\frac{1}{64}$	24	18	**13**	14	25	20
$10^{1.5}$	$\frac{1}{8}$	14	**11**	12	15	15	12
	$\frac{1}{16}$	17	14	**13**	15	19	15
	$\frac{1}{32}$	21	17	**14**	16	22	18
	$\frac{1}{64}$	27	21	**15**	17	27	21
100	$\frac{1}{8}$	17	**16**	17	19	19	17
	$\frac{1}{16}$	21	19	**18**	20	22	19
	$\frac{1}{32}$	26	22	**18**	21	26	21
	$\frac{1}{64}$	31	26	**19**	22	32	25
$10^{2.5}$	$\frac{1}{8}$	**25**	27	26	27	26	**25**
	$\frac{1}{16}$	27	28	26	27	28	**25**
	$\frac{1}{32}$	34	31	**27**	28	35	28
	$\frac{1}{64}$	42	35	**28**	29	42	33
1000	$\frac{1}{8}$	48	58	57	55	**47**	49
	$\frac{1}{16}$	**45**	56	53	49	46	**45**
	$\frac{1}{32}$	48	50	47	46	49	**41**
	$\frac{1}{64}$	56	53	47	47	57	**46**

It is again evidenced in our results for the two-phase Navier–Stokes problem that incorporating a further weighting into the boundary adjusted terms defining the LSC approach, that is utilising a norm dependent on the varying coefficients, can result in a lack of scalability of the preconditioner when refining the mesh; in particular, when the density and viscosity fields are piecewise constants or the interface is smoothed over a region depending on the mesh refinement. The detrimental effect again appears related to the interplay of the scaling used and the choice of damping parameter.

Table 6.9: Average preconditioned GMRES iterations after Newton linearisation when using the LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) using a fixed smoothed interface with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments. Iteration counts in bold show which method is most efficient.

Re	h	LSC(μ)	LSC($\mu^{1/2}$)	LSC($\mu, 1$)	LSC($\rho, 1$)	LSC(ρ, μ)	LSC($\rho, \mu^{1/2}$)
10	$\frac{1}{8}$	12	11	**10**	47	14	15
	$\frac{1}{16}$	15	12	**11**	13	17	13
	$\frac{1}{32}$	14	13	**11**	13	15	13
	$\frac{1}{64}$	15	14	**12**	13	15	14
$10^{1.5}$	$\frac{1}{8}$	14	13	**12**	50	16	17
	$\frac{1}{16}$	17	14	**13**	15	19	15
	$\frac{1}{32}$	16	15	**13**	16	17	15
	$\frac{1}{64}$	17	16	**14**	16	17	15
100	$\frac{1}{8}$	18	18	**17**	53	20	22
	$\frac{1}{16}$	21	19	**18**	20	22	19
	$\frac{1}{32}$	20	20	**18**	20	20	**18**
	$\frac{1}{64}$	21	21	**18**	20	20	19
$10^{2.5}$	$\frac{1}{8}$	**25**	28	26	54	27	29
	$\frac{1}{16}$	27	28	26	27	28	**25**
	$\frac{1}{32}$	27	28	26	28	27	**25**
	$\frac{1}{64}$	27	28	27	28	26	**25**
1000	$\frac{1}{8}$	48	59	57	66	**47**	50
	$\frac{1}{16}$	**45**	56	53	49	46	**45**
	$\frac{1}{32}$	40	49	46	44	39	**38**
	$\frac{1}{64}$	40	50	47	44	**39**	**39**

To explore this further, Table 6.10 compares values of the optimal inverse damping parameter ϵ^{-1} over all interface choices using the LSC($\mu, 1$) and LSC($\rho, \mu^{1/2}$) methods. As is consistent with the Stokes case, for LSC($\mu, 1$) the optimal ϵ^{-1} values roughly double each time we halve the grid size h. The exception to this is the high Reynolds number case of $Re = 1000$, however this may in part be due to the flow solution not being well resolved on the coarser meshes and the fact that the iteration counts initially drop somewhat.

Table 6.10: Optimal inverse damping parameter(s) ϵ^{-1} from $\epsilon^{-1} \in \{1, 2, 4, 8, \ldots, 64\}$ with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re, grid size h and choice of interface smoothing: PC denotes piecewise constant (no smoothing), S denotes a smoothed interface with $2\delta = 3h$, and FS denotes a smoothed interface of fixed width given by $2\delta = 3/16$.

		LSC$(\mu, 1)$			LSC$(\rho, \mu^{1/2})$		
Re	h	PC	S	FS	PC	S	FS
10	$\frac{1}{8}$	4	4	4	4, 8	2, 4, 8	4, 8, 16, 32
	$\frac{1}{16}$	8	4, 8, 16	4, 8, 16	4	4, 8	4, 8
	$\frac{1}{32}$	8, 16, 32	8	8, 16	4	4, 8	8
	$\frac{1}{64}$	16, 32, 64	16, 32	16, 32	4, 8	4, 8	8, 16, 32
$10^{1.5}$	$\frac{1}{8}$	2, 4, 8	4	2	4, 8	4	8, 16, 32
	$\frac{1}{16}$	4, 8	4, 8	4, 8	4	4, 8	4, 8
	$\frac{1}{32}$	8	8, 16	8	4, 8	4, 8	8, 16
	$\frac{1}{64}$	8, 16, 32	16	16	8	4, 8	16
100	$\frac{1}{8}$	2, 4	2, 4	2, 4	2	2, 4	2, 4, 8
	$\frac{1}{16}$	4	4, 8	4, 8	1, 2, 4	2, 4	2, 4
	$\frac{1}{32}$	4, 8	8	8	4	4	4, 8
	$\frac{1}{64}$	8	8, 16	8	4, 8	4, 8	8, 16
$10^{2.5}$	$\frac{1}{8}$	2	2	1, 2	1, 2	1, 2	2
	$\frac{1}{16}$	2	2	2	2	2	2
	$\frac{1}{32}$	4	4	4	2, 4	2, 4	4
	$\frac{1}{64}$	8	8	8	2, 4, 8	2, 4, 8	8
1000	$\frac{1}{8}$	1	1	1	1	2	1
	$\frac{1}{16}$	1	1, 2	1, 2	1, 2	1, 2	1, 2
	$\frac{1}{32}$	1	1, 2	1	1	1, 2	2
	$\frac{1}{64}$	2	2, 4	2, 4	2	2	2, 4

On the other hand, for LSC$(\rho, \mu^{1/2})$ the optimal inverse damping parameters vary less with h, though some increase is seen as we reduce h for the fixed smoothed interface case, which is consistent with the mesh-independence observed in this case. This provides further evidence that mesh-independence of the LSC approach is linked to optimal ϵ^{-1} values depending on h. We also observe from Table 6.10 that, for both methods, as the Reynolds number increases the optimal ϵ^{-1} slowly reduces to be closer to unity. This suggests that the damping parameter chosen should depend on the Reynolds number, though an optimal dependence is more difficult to quantify.

Our results illustrate that there is a delicate interplay between incorporating the variable coefficient nature of the problem into the LSC methodology, the imposition of boundary adjustments, and the rapidly varying density and viscosity fields which are inherent in two-phase flow problems. Indeed, both the scaling with the variable coefficient in the diagonalised mass matrices and the scaling used to de-emphasize certain contributions on the boundary are incorporated in effectively the same manner and so we may expect some interaction between these two scalings. In order to remove this detrimental coupling, and thus improve scalability, it appears necessary to incorporate only the boundary adjustments within the norm used for the least-squares problem and thus only weight the commutator using the variable coefficients.

Nonetheless, the above viewpoint restricts our ability to scale independently with both the density and viscosity to ameliorate the different terms present in F which may be useful when both the viscous and convective terms are important. We find evidence that utilising both coefficients in the scaling can be beneficial in our fixed smoothed viscosity case, where $\mathrm{LSC}(\rho, \mu^{1/2})$ proves favourable for the higher Reynolds number problems. Since the density and viscosity fields are sufficiently slowly varying comparative to the grid size h and do not depend on h in this case, the problem of negative interplay with the boundary adjustments is removed and, as such, scaling with both density and viscosity can be employed successfully.

Thus we see a more intricate set of rules determining the best choice of scaling in LSC based on the precise two-phase (or variable coefficient) nature of the problem, any mesh or interface refinement and the underlying Reynolds number of the flow, as well as the magnitude of the viscosity and density ratios as suggested in our Stokes results. Despite this, the choice $\mathrm{LSC}(\mu, 1)$ provides a good choice in all our results and thus we will use this as the LSC-type preconditioner of choice in later studies.

Before concluding our investigation into LSC-type preconditioning, we turn our attention to a more practical implementation of the LSC-type methods discussed. To do so we consider using inexact solves for the two Laplacian sub-problems required in the LSC inverse Schur complement approximation (6.34). We follow the same approach that we took for the (single) Laplacian sub-problem required in the PCD approach, as discussed in Section 5.4.1. As such, we use a single V-cycle of an AMG method, specifically as is provided by the `HSL_MI20` code [29], to approximate the solution to each of the two Laplacian sub-problems required. Note that the two sub-problems are different and so the approach is applied separately for each one. We recall that such choices constitute the default behaviour in IFISS [209].

Table 6.11 compares the LSC$(\mu, 1)$ and LSC$(\rho, \mu^{1/2})$ methods implemented in both the ideal and more practical AMG iterated cases. Again, we have emboldened which of the two methods requires the fewest iterations in each problem instance, this time only for the AMG iterated cases; values for the ideal implementation are provided for direct comparison. We see a broadly similar situation to the ideal implementation in terms of LSC$(\mu, 1)$ being mostly favourable until the high Reynolds number case where LSC$(\rho, \mu^{1/2})$ becomes more efficient. However, there is a messier nature to the results in Table 6.11 since both methods are affected by the poorer performance of the AMG approximations to the Laplacian sub-problems which produces iteration counts that are a little more erratic. While, for the most part, the AMG iterated implementation is not significantly worse than the ideal implementation, there are some instances where the iteration counts are notably larger and these are typically for the finer grids used. Furthermore, all instances of mesh-independent behaviour previously observed in the ideal case are corrupted due to an overall lack of scalability from the AMG iterated implementation.

In general, our results suggest that the AMG technique used can struggle to well approximate the solution of the Laplacian sub-problems required. Since this is not an issue in the single-phase case, we believe the level of deterioration seen is due to the highly varying scaling incorporated in the Laplacian-type operators. Moreover, our experience suggests that the difficulty and somewhat erratic behaviour is only prominent for the discretisation choices used in the LSC approach, namely stemming from Laplacian terms of the form $BD^{-1}B^T$ rather than the choice of A_p in (5.32) for the PCD approach. We have also observed a preference for the original boundary adjustment technique of [72] in the AMG iterated case; this may be because this approach affects less components on the boundary (only tangential components) and so gives less variability in the scaling which features in one of the Laplacian sub-problems. Thus, in the practical case, there may be a trade-off between scaling with the two-phase coefficients and boundary adjustments, and minimising the variability of such scaling so as to aid the AMG method used.

The exploration of appropriate approximations to the Laplacian sub-problems which feature in LSC-type methods for two-phase flow is an important outstanding issue. We have not pursued the consideration of AMG methodologies other than the `HSL_MI20` implementation present in IFISS and then only with the use of a single V-cycle. It may be that additional cycles or alternative methods can be applied more successfully and such work is an important direction of future research if the LSC-type approach is to yield effective solvers for large-scale two-phase flow problems.

Table 6.11: Average preconditioned GMRES iterations after Newton linearisation when using the LSC-type preconditioner (6.19), implemented in the ideal / AMG iterated case, with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2\times10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8\times10^{-2}$ (values for air-water flow), and varying Reynolds number Re, grid size h, and choice of interface smoothing: PC denotes piecewise constant (no smoothing), S denotes a smoothed interface with $2\delta = 3h$, and FS denotes a smoothed interface of fixed width given by $2\delta = 3/16$. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments. Iteration counts in bold show which AMG iterated method is most efficient for each choice of interface smoothing. We note that AMG iterated results for the other choices of scaling in the LSC-type methods we consider are given in Tables A.2 to A.4.

		LSC$(\mu, 1)$			LSC$(\rho, \mu^{1/2})$		
Re	h	PC	S	FS	PC	S	FS
10	$\frac{1}{8}$	10 / **13**	10 / 11	10 / **10**	12 / 16	11 / **10**	15 / 18
	$\frac{1}{16}$	11 / **12**	11 / **11**	11 / **11**	14 / 16	13 / 14	13 / 14
	$\frac{1}{32}$	13 / **13**	11 / **18**	11 / **13**	17 / 17	16 / **18**	13 / 17
	$\frac{1}{64}$	14 / **17**	13 / **21**	12 / **19**	21 / 20	20 / 27	14 / **19**
$10^{1.5}$	$\frac{1}{8}$	13 / **14**	12 / **12**	12 / **13**	14 / 18	12 / **12**	17 / 20
	$\frac{1}{16}$	14 / **15**	13 / **13**	13 / **13**	16 / 17	15 / 16	15 / 16
	$\frac{1}{32}$	15 / **15**	14 / **19**	13 / **16**	19 / 20	18 / 20	15 / 18
	$\frac{1}{64}$	17 / **19**	15 / **24**	14 / **18**	22 / 22	21 / 26	15 / 21
100	$\frac{1}{8}$	17 / **19**	17 / 19	17 / **17**	18 / 21	17 / **17**	22 / 24
	$\frac{1}{16}$	18 / **20**	18 / 22	18 / 22	21 / 21	19 / **20**	19 / **20**
	$\frac{1}{32}$	20 / **22**	18 / **23**	18 / **20**	23 / 23	21 / 24	18 / 22
	$\frac{1}{64}$	21 / 28	19 / **26**	18 / 25	27 / **26**	25 / 30	19 / **24**
$10^{2.5}$	$\frac{1}{8}$	26 / **28**	26 / 26	26 / **26**	25 / **28**	25 / **25**	29 / 32
	$\frac{1}{16}$	27 / 29	26 / **27**	26 / **27**	27 / **27**	25 / **27**	25 / **27**
	$\frac{1}{32}$	28 / 32	27 / 33	26 / **29**	31 / **31**	28 / **30**	25 / **29**
	$\frac{1}{64}$	30 / **32**	28 / **35**	27 / **33**	36 / 36	33 / 40	25 / 34
1000	$\frac{1}{8}$	59 / 60	57 / 57	57 / 57	49 / **52**	49 / **50**	50 / **54**
	$\frac{1}{16}$	57 / 60	53 / 59	53 / 59	50 / **51**	45 / **44**	45 / **44**
	$\frac{1}{32}$	48 / 53	47 / 55	46 / 54	44 / **51**	41 / **46**	38 / **45**
	$\frac{1}{64}$	49 / 55	47 / 55	47 / 55	49 / **51**	46 / **54**	39 / **45**

Furthermore, tuning of the damping parameter ϵ may be crucial for such problems in order to gain scalable behaviour. Our results on a model test problem indicate this may also be a challenging matter, particularly if the Reynolds numbers encountered during a simulation are not known in advance and change significantly over the course of the simulation. Nonetheless, we have laid down some groundwork in understanding such potential issues and proposed choices of scaling within the LSC methodology which can help provide more efficient preconditioners in the two-phase scenario.

We conclude this chapter by noting an empirical observation on the behaviour of LSC preconditioner for simplex elements in the single-phase case. While mesh-independence of the LSC approach applied to problems discretised using quadrilateral elements has been demonstrated, it has been observed (though not evidenced in published work) that performance is poorer when utilising simplex elements [152]. Primarily, mesh-independence is lost and scaling is similar to the earlier (unscaled) BFBt approach in [67] (which effectively uses the diagonal scaling matrix $D = I$). To the best of our knowledge, there is currently no clear view or consensus as to whether this deterioration for simplex elements is inherent and unavoidable in LSC-type approaches or whether, through a correct choice of scaling matrix or otherwise, this issue can be bypassed in order to gain mesh-independent behaviour on simplex meshes. As such, questions on the utility of LSC in such cases remains unresolved.

Chapter 7

Comparison of block preconditioners for two-phase flow

In this chapter we provide example results for the two model test problems described in Problem 1 and Problem 2 of Section 2.5 and compare the performance of different block preconditioners. Firstly, we compare the two approaches we have developed and adapted for two-phase flow, namely the PCD and LSC preconditioners. We then go on to make comparisons with other block preconditioners from the literature.

7.1 A comparison of PCD and LSC

Before detailing numerical results, we first make some comparisons of PCD and LSC based on the underlying approaches, their applicability, and the resulting form of Schur complement preconditioner. We note that detailed comparisons between the single-phase PCD and LSC approaches are given in [165] and [71]. Predominantly, our two-phase strategies compare similarly.

We summarise the key differences: PCD requires the construction of additional matrices based on the pressure space operators while LSC requires only matrices that are readily available apart from scaled velocity mass matrices which are easily constructed. This further means that LSC directly applies when a discontinuous pressure approximation is used while the formulation of PCD is more involved.

On the other hand, PCD naturally extends to the case of stabilised elements while LSC requires appropriate stabilisation terms to be constructed in this case. Additionally, only one solution of a discrete Laplacian sub-problem is needed by PCD while LSC requires two and so is approximately twice as expensive to apply in terms of the Schur complement approximation.

Boundary conditions are important for both preconditioners, though typically more so for the PCD approach. However, for inflow and outflow as well as enclosed flow problems we have a good understanding of appropriate boundary conditions. For PCD the issue of boundary conditions for the pressure space operators arise naturally and can be incorporated into their discretisation. The consideration of boundary conditions does not explicitly arise within the LSC approach but they can be incorporated in the choice of norm used to formulate the least-squares problem. However, a damping parameter must be chosen to de-emphasise certain components of the commutator on the boundary. While for the single-phase case a good choice is known, for two-phase flow the choice can become more important and it is not fully clear how to determine the best choice. Nonetheless, reasonable performance can still be obtained.

In order to gain some appreciation of how important it is to take into account appropriate scaling, we first compare with the single-phase methods. Our comparison is taken from results found in [27] which are reproduced in Tables 7.1 and 7.2. Here we display results for a variety of density ratios $\widehat{\rho} = \rho_2/\rho_1$ and viscosity ratios $\widehat{\mu} = \mu_2/\mu_1$ with given Reynolds number $Re = 100$ and grid size $h = 1/128$ for the steady cavity problem defined in Problem 1. We make use of $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ finite elements and consider Picard linearisation.

Table 7.1 displays results for the split two-phase PCD preconditioner as well as the LSC(μ) approach without any adjustments for boundary conditions. We observe that, while there is some variation with the density ratio and, in particular, the viscosity ratio, the performance is fairly robust across the wide range of ratios tested. This is not true of the single-phase PCD or LSC preconditioners for this problem; these methods give performance which depends significantly on the density and viscosity ratios, and poor mesh-dependence away from ratios close to unity. The equivalent values to those in Table 7.1 reach into several hundreds of iterations for the single-phase methods; these results are shown in Table 7.2. As such, we see that it is critical that the two-phase nature of the problem be incorporated into the PCD and LSC preconditioners.

We note that some variation is also seen in the results for the generalised Cahouet–Chabard preconditioner in [161]; in particular, for large ratios in opposite directions for density and viscosity the Schur complement approximation becomes much less efficient. The results given in [100], for an augmented Lagrangian approach to solve the Navier–Stokes equations with variable viscosity, also depend on the viscosity through the minimum and maximum values taken in the domain.

Table 7.1: Average preconditioned GMRES iterations after Picard linearisation when using split two-phase PCD / LSC(μ) for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) with $Re = 100$, $h = 1/128$, and varying density ratio $\widehat{\rho} = \rho_2/\rho_1$ and viscosity ratio $\widehat{\mu} = \mu_2/\mu_1$. Values marked "$\star$" are omitted since here the dominating Reynolds number changes, being associated with the second phase.

$\widehat{\mu}$ \ $\widehat{\rho}$	10^{-3}	10^{-2}	10^{-1}	1	10	100	1000
10^{-3}	30 / 59	$\star$	$\star$	$\star$	$\star$	$\star$	$\star$
10^{-2}	29 / 58	30 / 58	$\star$	$\star$	$\star$	$\star$	$\star$
10^{-1}	24 / 54	24 / 54	25 / 54	$\star$	$\star$	$\star$	$\star$
1	19 / 38	19 / 38	19 / 38	20 / 40	$\star$	$\star$	$\star$
10	24 / 44	24 / 44	24 / 44	24 / 44	27 / 44	$\star$	$\star$
100	26 / 38	26 / 38	26 / 38	27 / 38	27 / 38	29 / 37	$\star$
1000	26 / 36	26 / 36	26 / 36	27 / 36	27 / 36	27 / 34	28 / 33

Table 7.2: Average preconditioned GMRES iterations after Picard linearisation when using single-phase PCD / single-phase LSC for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) with $Re = 100$, $h = 1/128$, and varying density ratio $\widehat{\rho} = \rho_2/\rho_1$ and viscosity ratio $\widehat{\mu} = \mu_2/\mu_1$. Values marked "$\star$" are omitted since here the dominating Reynolds number changes, being associated with the second phase.

$\widehat{\mu}$ \ $\widehat{\rho}$	10^{-3}	10^{-2}	10^{-1}	1	10	100	1000
10^{-3}	173 / 261	$\star$	$\star$	$\star$	$\star$	$\star$	$\star$
10^{-2}	104 / 74	101 / 74	$\star$	$\star$	$\star$	$\star$	$\star$
10^{-1}	46 / 38	46 / 38	45 / 38	$\star$	$\star$	$\star$	$\star$
1	29 / 37	29 / 37	29 / 37	30 / 39	$\star$	$\star$	$\star$
10	36 / 36	36 / 36	36 / 36	36 / 36	43 / 36	$\star$	$\star$
100	76 / 81	76 / 81	76 / 81	76 / 81	76 / 80	88 / 71	$\star$
1000	125 / 320	125 / 320	125 / 320	125 / 320	123 / 322	120 / 310	140 / 296

For the remaining numerical results in this section we use the scaling provided in the LSC($\mu, 1$) method since this provided the best performance overall in the results of Chapter 6. In order to give a more realistic choice for boundary adjustments, we will always select the inverse of the damping parameter to be $\epsilon^{-1} = 10$ (as in [71]) as opposed to it being selected optimally by trial and error. This choice still provides good results over the range of Reynolds numbers and grids used. For the two-phase PCD approach, subsequent results use the inverse Schur complement approximation in (5.79) derived from the bi-weighted commutator but will use a split implementation as in (5.92) which was shown to be favourable in certain circumstances in Section 5.5.2.

Table 7.3: Average preconditioned GMRES iterations after Newton linearisation when using two-phase PCD / LSC($\mu, 1$) for the $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) using a smoothed interface with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), and varying Reynolds number Re and grid size h.

$h \setminus Re$	10	$10^{1.5}$	100	$10^{2.5}$	1000
$\frac{1}{16}$	17 / 39	21 / 43	27 / 55	38 / 87	71 / 144
$\frac{1}{32}$	17 / 32	21 / 46	28 / 48	35 / 73	68 / 156
$\frac{1}{64}$	18 / 39	22 / 44	28 / 60	37 / 82	58 / 140
$\frac{1}{128}$	18 / 45	22 / 49	29 / 57	39 / 75	59 / 124
$\frac{1}{256}$	18 / 42	23 / 45	30 / 50	40 / 68	60 / 96

Table 7.4: Average preconditioned GMRES iterations after Newton linearisation when using two-phase PCD / LSC($\mu, 1$) for the $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ solution to the steady lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), and varying Reynolds number Re and grid size h.

$h \setminus Re$	10	$10^{1.5}$	100	$10^{2.5}$	1000
$\frac{1}{16}$	18 / 12	22 / 14	31 / 20	47 / 35	111 / 68
$\frac{1}{32}$	18 / 11	23 / 15	31 / 21	44 / 34	96 / 62
$\frac{1}{64}$	18 / 13	23 / 15	32 / 20	41 / 32	85 / 62
$\frac{1}{128}$	18 / 14	23 / 16	32 / 21	42 / 31	65 / 57
$\frac{1}{256}$	18 / 16	23 / 18	32 / 22	42 / 32	64 / 59

Tables 7.3 to 7.8 detail results for the cavity and step problems in both the steady and time-dependent cases and between them utilise the three different choices of finite elements that we have considered. We remark that for $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ elements we have found that the AMG solver we use in IFISS for the Laplacian sub-problems in the LSC preconditioner can fail. Thus, in this case we utilise an ideal implementation of the LSC preconditioner; all other instances use AMG iterated implementations of the inverse Schur complement approximation. In each of our examples, we utilise density and viscosity ratios which correspond to air–water flow (namely $\widehat{\rho} = 1.2 \times 10^{-3}$ and $\widehat{\mu} = 1.8 \times 10^{-2}$) but note that similar trends are also seen when other ratios are used, as suggested in the results of Table 7.1.

Table 7.5: Average preconditioned GMRES iterations after Newton linearisation when using two-phase PCD / LSC$(\mu, 1)$ for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady backward-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), and varying Reynolds number Re and grid size h.

$h \setminus Re$	10	100	500
$\frac{1}{4}$	23 / 22	27 / 27	58 / 76
$\frac{1}{8}$	23 / 14	26 / 24	52 / 74
$\frac{1}{16}$	24 / 15	27 / 24	47 / 70
$\frac{1}{32}$	25 / 16	28 / 24	46 / 67
$\frac{1}{64}$	25 / 17	30 / 25	48 / 70

Table 7.6: Average preconditioned GMRES iterations after Newton linearisation when using two-phase PCD / LSC$(\mu, 1)$ for the $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), time-step $\Delta t = 1/10$, and varying Reynolds number Re and grid size h.

$h \setminus Re$	10	$10^{1.5}$	100	$10^{2.5}$	1000
$\frac{1}{16}$	12 / 32	13 / 29	14 / 32	17 / 34	25 / 38
$\frac{1}{32}$	13 / 33	13 / 27	14 / 28	16 / 30	21 / 35
$\frac{1}{64}$	14 / 32	14 / 35	14 / 32	16 / 33	19 / 34
$\frac{1}{128}$	14 / 38	14 / 37	14 / 32	16 / 33	18 / 32
$\frac{1}{256}$	15 / 38	15 / 41	15 / 42	16 / 39	18 / 38

We first compare dependence on the Reynolds number Re. In the steady case, from Tables 7.3 to 7.5 we see that both the PCD and LSC approaches depend fairly mildly on the Reynolds number with larger iteration counts seen for higher Reynolds numbers. However, in the time-dependent case, as illustrated in Tables 7.6 to 7.8, the iteration counts are broadly similar across the range of Reynolds numbers used (perhaps with the exception of LSC$(\mu, 1)$ for the results in Table 7.8) and sometimes, if the time-step is small enough, can slowly decrease as Re increases. These observations are consistent with the single-phase case (see, for example, [71]). We further note that, as expected, as we decrease the time-step both methods require fewer iterations since the problem becomes dominated by the contribution from the mass matrix.

Table 7.7: Average preconditioned GMRES iterations after Picard linearisation when using two-phase PCD / LSC$(\mu, 1)$ for the $\boldsymbol{Q}_2$–$\boldsymbol{P}_{-1}$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), $h = 1/128$, and varying Reynolds number Re and time-step Δt.

$\Delta t \setminus Re$	10	$10^{1.5}$	100	$10^{2.5}$	1000
10	17 / 15	21 / 18	25 / 19	31 / 21	38 / 27
1	17 / 14	18 / 14	20 / 13	22 / 14	23 / 17
$\frac{1}{10}$	14 / 10	14 / 11	15 / 12	15 / 14	17 / 17

Table 7.8: Average preconditioned GMRES iterations after Newton linearisation when using two-phase PCD / LSC$(\mu, 1)$ for the $\boldsymbol{Q}_1$–$\boldsymbol{Q}_1$ solution to the backward-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), $h = 1/128$, and varying Reynolds number Re and time-step Δt.

$\Delta t \setminus Re$	10	100	500
10	24 / 30	28 / 37	28 / 51
1	23 / 28	21 / 32	18 / 39
$\frac{1}{10}$	19 / 26	16 / 28	15 / 31

Next we turn to consider the dependence on the grid size h, collating results from Tables 7.3 to 7.6. The two-phase PCD preconditioner shows good performance with effectively mesh-independent behaviour in all our results. On the other hand, our results for the LSC approach typically show mesh-dependence with iteration counts mildly deteriorating as we refine the grid. This is to be expected based on our observations in Chapter 6 for the AMG iterated implementation and the fact that we use a fixed damping parameter ϵ for boundary adjustments, independent of the grid.

Overall, while the LSC approach sometimes requires fewer iterations to achieve convergence compared with the two-phase PCD preconditioner, the fact that it is approximately twice as expensive to apply means that the two-phase PCD method is almost always more efficient. While additional work is required to build the pressure space operators in PCD, the superior mesh-dependence and lack of any parameters which need tuning ensures that two-phase PCD is much more favourable within the scope covered by our work. Furthermore, PCD requires no modification for use with stabilised elements and can be used with discontinuous pressure approximations very effectively, as demonstrated in Table 7.4 and previously in Table 5.3.

7.2 Comparisons with other block preconditioners

In this section we consider two other block preconditioners which can be applied in our case of two-phase flow. We first consider an alternative LSC-type method, namely the $\text{LSC}_{\text{diag}(F)}$ approach discussed in [150, 230] which uses the matrix $D = \text{diag}(F)$ as the diagonal scaling matrix D in (6.13). Results in Table 7.9 are taken from [27] where we compared $\text{LSC}_{\text{diag}(F)}$ with $\text{LSC}(\mu)$ without any boundary adjustments being incorporated. We observe a significant dependence on h in the performance of $\text{LSC}_{\text{diag}(F)}$ so that, on the finest mesh used, more than triple the number of iterations is required compared with $\text{LSC}(\mu)$. The much poorer scalability seen in $\text{LSC}_{\text{diag}(F)}$ shows the importance of the scaling matrix D which features in LSC-type methods. Our considerations of weighted commutators suggested that it is much more beneficial to make use of an appropriately weighted mass matrix in order to define D, as supported by results in Chapter 6.

Table 7.9: Average preconditioned GMRES iterations after Picard linearisation when using $\text{LSC}_{\text{diag}(F)}$ / $\text{LSC}(\mu)$ without boundary adjustments for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), and varying Reynolds number Re and grid size h.

$h \setminus Re$	10	$10^{1.5}$	100	$10^{2.5}$	1000
$\frac{1}{8}$	21 / 15	23 / 15	26 / 18	29 / 24	41 / 34
$\frac{1}{16}$	28 / 15	30 / 19	32 / 23	37 / 27	51 / 37
$\frac{1}{32}$	54 / 18	57 / 20	63 / 28	73 / 34	89 / 30
$\frac{1}{64}$	91 / 24	95 / 26	103 / 41	117 / 33	148 / 46
$\frac{1}{128}$	145 / 48	154 / 51	166 / 56	192 / 62	244 / 67

The second block preconditioner that we make comparisons with is the SIMPLE-like preconditioner (4.18). This method can be straightforwardly applied to two-phase flow since it is purely algebraic in nature and uses components directly from the saddle point matrix (4.1). These features make it a convenient choice which is broadly applicable to many saddle point systems. We remark that the Schur complement approximation $\widehat{S} = B\text{diag}(F)^{-1}B^T + C$ for incompressible flow problems corresponds to a scaled Laplacian operator. As such, for a practical implementation we use a single V-cycle of an AMG solver identically to the way in which we treat the Laplacian sub-problems within the PCD and LSC approach. Note that this means the cost in applying the SIMPLE-like preconditioner is comparable with the PCD method.

Table 7.10: Average preconditioned GMRES iterations after Newton linearisation when using two-phase PCD / SIMPLE-like preconditioning for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), and varying Reynolds number Re and grid size h.

$h \setminus Re$	10	$10^{1.5}$	100	$10^{2.5}$	1000
$\frac{1}{4}$	15 / 25	18 / 27	24 / 31	35 / 40	85 / 84
$\frac{1}{8}$	16 / 39	19 / 42	24 / 49	33 / 65	70 / 111
$\frac{1}{16}$	18 / 58	21 / 63	27 / 72	34 / 90	58 / 167
$\frac{1}{32}$	19 / 110	23 / 117	29 / 129	36 / 157	56 / 243
$\frac{1}{64}$	20 / 180	24 / 192	31 / 210	38 / 247	58 / 354

Table 7.11: Average preconditioned GMRES iterations after Newton linearisation when using LSC$(\rho, \mu^{1/2})$ / SIMPLE-like preconditioning for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air–water flow), grid size $h = 1/64$, and varying Reynolds number Re and time-step Δt.

$\Delta t \setminus Re$	10	$10^{1.5}$	100	$10^{2.5}$	1000
10	25 / 179	23 / 192	31 / 207	38 / 233	43 / 294
1	24 / 177	20 / 185	24 / 188	24 / 190	21 / 190
$\frac{1}{10}$	20 / 164	15 / 154	18 / 131	16 / 116	10 / 109

In Table 7.10 we compare the two-phase PCD and SIMPLE-like approaches for the steady cavity problem. It is clear that the SIMPLE-like preconditioner provides a much poorer approximation and lacks scalability. When compared with PCD, the SIMPLE-like approach requires up to nine times more iterations on the finest grid used. Similar conclusions can be drawn in the time-dependent case, as evidenced in Table 7.11 where we now compare the SIMPLE-like preconditioner with LSC$(\rho, \mu^{1/2})$ using $\epsilon^{-1} = 2$. Here we see that more than ten times the iterations are required by the SIMPLE-like method compared with LSC$(\rho, \mu^{1/2})$ for the combination of the smallest time-step and largest Reynolds number used.[1] Overall, these results emphasise the central importance of developing effective Schur complement approximations in order to obtain efficient and scalable block preconditioners of the form (4.11).

[1]We also note that, with small time-steps, LSC$(\rho, \mu^{1/2})$ shows an improved trend in Reynolds number compared with results for LSC$(\mu, 1)$ in Table 7.7, suggesting that scaling with ρ may be beneficial in the time-dependent case as the dominant term becomes the density scaled time-stepping term. Further study is required in order to ascertain which scaling of LSC is most beneficial here.

Chapter 8

Multipreconditioning using sMPGMRES

In challenging fluid flow simulations used to model hydraulic processes it is not always clear what the best choice of preconditioner might be for solving a given linear system. Moreover, during a simulation different flow regimes may be encountered and the optimal choice from a set of preconditioners may change throughout the computation. One can imagine adaptively changing the preconditioner based on tracking the current flow regime. However, this requires knowing *a priori* which preconditioner is likely best in any given regime as well as a suitable evaluation of the current flow, which may well vary within the domain. The required sophistication and good prior knowledge of the preconditioners' performance makes such an adaptive approach less appealing.

Instead we might consider using multiple preconditioners simultaneously, aiming to get the best of each. If we can combine the preconditioners then we would like to know whether we can achieve performance similar to using the (unknown) best preconditioner and, further, if together they provide an improvement over any individual one. Another key question to ask would be that of robustness: whether inclusion of a poorly performing preconditioner significantly affects the overall performance.

These ideas are encompassed within multipreconditioning strategies, where the iterative method or preconditioning choice incorporates using more than one preconditioner. There are several ways in which multipreconditioning can be employed but in each it is important to consider any additional computational cost incurred weighed against the performance improvements that can be gained in the iterative solver. Note, however, that such a strategy might not simply be aiming to give the optimal performance for solving a given system but also to provide an overall robustness during a simulation when it is not known whether a preconditioner will perform well across the spread of regimes encountered in the simulation.

We will focus on a recent strategy of Greif, Rees, and Szyld [88] which yields a multipreconditioned GMRES algorithm and, in particular, a selective variant which limits the computation cost involved. Further, there is good scope for parallelisation, with the application of each preconditioner carried out in parallel. This reduces the overhead of simultaneously incorporating several preconditioners. Before detailing this approach we outline and review other methodology for multipreconditioning.

8.1 Multipreconditioning strategies

One simple way to incorporate multiple preconditioners into an iterative method is to change the preconditioner used at each iteration, in which case a flexible solver such as Saad's FGMRES [193] is required that allows for a varying preconditioner. This is exemplified in *cycling*, where the preconditioner used at each iteration is changed in a prescribed order in a cyclic manner. This idea was proposed by Rui, Yong, and Chen [192], where they use it to solve electromagnetic wave scattering problems. Their results demonstrated convergence never better than the best choice of preconditioner on its own; such a choice not being known in advance however. While only observed numerically, it stands to reason that such an approach is unlikely to provide an improvement over the best preconditioner for any given linear system, though it might help provide robustness over an unknown spectrum of problems.

Another strategy is to form a single preconditioner from the options available and use this at each iteration. This is employed in *combination* preconditioning, in which the action of the inverse of the preconditioner is a linear combination of the individual actions; for instance, with two preconditioners ($\mathcal{P}_1$ and $\mathcal{P}_2$) given by

$$\mathcal{P}^{-1} = \alpha_1 \mathcal{P}_1^{-1} + \alpha_2 \mathcal{P}_2^{-1}, \tag{8.1}$$

for some constants α_1 and α_2. The term was introduced by Stoll and Wathen [214], who showed how two preconditioners, which are self-adjoint with respect to different inner products, can be combined to yield further preconditioners which are self-adjoint in a potentially new inner product. The main consideration revolves around maintaining symmetry, in some nonstandard inner product, so that a CG-type or MINRES-type method can be employed. Pestana and Wathen [177] further build on these ideas, in particular looking to ensure positive definiteness and so enable the use of the more efficient CG-type method, which they show can be gained in certain circumstances even if the individual preconditioners are not positive definite in their respective inner products. While examples given in these works focus on obtaining

some desired structure, the form considered in combination preconditioning could equally be applied in the more general nonsymmetric case with less restriction on parameter choice and need for a nonstandard inner product.

A somewhat similar idea, using linear combinations of preconditioned operators, is found in the earlier *multi-splitting* algorithm of O'Leary and White [157]. Here the idea is phrased in terms of combining multiple different splitting methods to solve a linear system, hence the name. It can also be thought of as a stationary iteration with each splitting providing a preconditioner, hence a multipreconditioning algorithm. More formally, for given preconditioners $\mathcal{P}_i$, $i = 1, \ldots, \ell$ and positive semi-definite diagonal weighting matrices $\mathcal{D}_i$ such that $\sum_i \mathcal{D}_i = I$, then the multi-splitting approach uses the stationary iteration

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \sum_{i=1}^{\ell} \mathcal{D}_i \mathcal{P}_i^{-1} \boldsymbol{r}^{(k)}. \tag{8.2}$$

As with combination preconditioning, the weights $\mathcal{D}_i$ (or α_i in (8.1)) must be chosen in advance and fixed during every iteration. This limitation on the weights involved in linear combinations is not present in the multipreconditioned GMRES algorithm we will consider.

We note the scope for parallelism in these approaches. In both the combination preconditioning and multi-splitting methods the action of the preconditioners can be performed separately, thus in parallel, before we sum together the contributions. On the other hand, cycling is a serial process in which the preconditioners cannot be used simultaneously, though only one application is needed per iteration. Each of these methods then have a cost equivalent to their respective underlying iterative methods, so long as the simultaneous action of the preconditioners (if needed) are computed in parallel. Nonetheless, the performance of the iterative method will depend on the overall effectiveness of the preconditioners and how they are combined.

We will consider a multipreconditioned GMRES algorithm; this retains the parallelisable application of preconditioners but computes weights as part of the algorithm which are in some sense optimal. The same idea was first applied to the conjugate gradient method for symmetric positive definite problems by Bridson and Greif [38] to give *multipreconditioned conjugate gradients* (MPCG). To do so, one considers not just a single new search direction at each iteration but several, given by applying different preconditioners, combining them to preserve an optimality condition in the energy norm. Unfortunately, a drawback of this method over the original CG method is that we lose the favourable short-term recurrence relation which allows CG to be

particularly efficient. However, results in [38] suggest that often it is acceptable to truncate the A-conjugation step required to the standard short recurrence. Since, in general, we do not have short recurrences in GMRES this is not an issue. Nonetheless, when using multiple preconditioners the search space grows exponentially fast as we iterate, as such a selective approach to restrict this growth to be linear is necessary.

8.2 Multipreconditioned GMRES (MPGMRES)

In the standard preconditioned GMRES (or FGMRES) method, at each iteration a new search direction, based on the preconditioned operator, is added to the search space and a least-squares problem is solved to find a solution with minimum residual norm. The key idea behind the *multipreconditioned GMRES* (MPGMRES) method of Greif, Rees, and Szyld [88] is to add *multiple* new search directions at each iteration stemming from the different preconditioned operators available. In fact, the method adds all new directions from combinations of the preconditioned operators applied to vectors in the current search space, making the search space very rich. An Arnoldi-type block procedure is then used to obtain an orthonormal basis of the search space. MPGMRES then computes the optimal new iterate from this space in the minimum residual least-squares sense. Hence, note that the weights defining the contributions from each preconditioned operator are computed as part of the procedure, unlike in the aforementioned multipreconditioning strategies (aside from MPCG).

To understand how this *complete* MPGMRES algorithm works, suppose we have $\ell \geq 2$ preconditioners $\mathcal{P}_i$, $i = 1, \ldots, \ell$. We start with an initial residual vector $\boldsymbol{r}^{(0)}$, which we normalise to give the first basis vector $V^{(1)} = \beta^{-1}\boldsymbol{r}^{(0)}$, with $\beta = \|\boldsymbol{r}^{(0)}\|_2$, and collect together the preconditioned (normalised) residuals

$$Z^{(1)} = \beta^{-1}\left[\mathcal{P}_1^{-1}\boldsymbol{r}^{(0)}, \ldots, \mathcal{P}_\ell^{-1}\boldsymbol{r}^{(0)}\right] \in \mathbb{R}^{n\times\ell}. \tag{8.3}$$

Using an Arnoldi-type block procedure[1] we orthogonalise every column of $W = \mathcal{A}Z^{(1)}$ with respect to our current basis $V^{(1)}$ and amongst themselves by using a reduced QR factorisation. Normalising then provides new orthonormal basis vectors $V^{(2)} \in \mathbb{R}^{n\times\ell}$.

At each iteration, k, we increase the search space by applying each preconditioner to our newest basis vectors $V^{(k)}$, computing

$$Z^{(k)} = \left[\mathcal{P}_1^{-1}V^{(k)}, \ldots, \mathcal{P}_\ell^{-1}V^{(k)}\right] \in \mathbb{R}^{n\times\ell^k}. \tag{8.4}$$

[1]For further details, including discussion of implementation, see [88, p. 218] and references therein.

The Arnoldi-type block procedure is then used to orthogonalise $W = \mathcal{A}Z^{(k)}$ with respect to the current basis $\widetilde{V}_k = \begin{bmatrix} V^{(1)} & \dots & V^{(k)} \end{bmatrix}$ and within itself. This yields new basis vectors $V^{(k+1)} \in \mathbb{R}^{n \times \ell^k}$ and, by storing the coefficients from the Arnoldi-type step in a block upper Hessenberg matrix $\widetilde{H}_k$, an Arnoldi-type decomposition

$$\mathcal{A}\widetilde{Z}_k = \widetilde{V}_{k+1}\widetilde{H}_k, \tag{8.5}$$

where $\widetilde{Z}_k = \begin{bmatrix} Z^{(1)} & \dots & Z^{(k)} \end{bmatrix}$ and

$$\widetilde{H}_k = \begin{bmatrix} H^{(1,1)} & H^{(1,2)} & \cdots & H^{(1,k)} \\ H^{(2,1)} & H^{(2,2)} & & H^{(2,k)} \\ & \ddots & & \vdots \\ & & H^{(k,k-1)} & H^{(k,k)} \\ & & & H^{(k+1,k)} \end{bmatrix}. \tag{8.6}$$

Note that the subdiagonal blocks $H^{(i+1,i)}$ of $\widetilde{H}_k$ are all upper triangular, stemming from the reduced QR factorisation, and $H^{(1,i)} \in \mathbb{R}^{1 \times \ell}$ so that the top blocks are just row vectors, hence the matrix $\widetilde{H}_k$ is upper Hessenberg (not just block upper Hessenberg). The number of basis vectors (columns) in $\widetilde{V}_{k+1}$ is

$$c_k = \sum_{i=0}^{k} \ell^i = \frac{\ell^{k+1} - 1}{\ell - 1}, \tag{8.7}$$

while the number of columns in $\widetilde{Z}_k$ is

$$c_k - 1 = \ell c_{k-1} = \frac{\ell(\ell^k - 1)}{\ell - 1}, \tag{8.8}$$

and hence $\widetilde{H}_k \in \mathbb{R}^{c_k \times (c_k - 1)}$. As such, there is an exponential increase in the size of the search space, and associated matrices, every iteration of the complete MPGMRES algorithm.

Remark 7 (Linear dependency) *In the above, we have an implicit assumption that both $Z^{(k)}$ and $\widetilde{Z}_k$ are of full rank. If we have linear dependence, because of redundancy in the user-provided preconditioners, a deflation-type idea can be used to remove this issue; see [88, §3]. We maintain this implicit assumption on the understanding that, should it not hold, deflation will be used to mitigate the problem and the appropriate matrices reduced accordingly.*

Similarly to FGMRES, we solve a linear least-squares problem for the minimum residual solution to

$$\min_{\boldsymbol{x} \in \boldsymbol{x}^{(0)} + \operatorname{range}\left(\widetilde{Z}_k\right)} \|\boldsymbol{b} - \mathcal{A}\boldsymbol{x}\|_2 = \min_{\boldsymbol{y}} \left\| \|\boldsymbol{r}^{(0)}\|_2 \, \boldsymbol{e}_1 - \widetilde{H}_k \boldsymbol{y}, \right\|_2, \tag{8.9}$$

where $\boldsymbol{x} = \boldsymbol{x}^{(0)} + \widetilde{Z}_k \boldsymbol{y}$. Note that there is a natural generalisation of the standard GMRES polynomial minimisation property. While GMRES minimises a (univariate) polynomial of the preconditioned operator acting on the initial residual, MPGMRES minimises a multivariate polynomial in ℓ *non-commuting* variables. That is, defining this space of all such multivariate polynomials of degree k (or less) as $\mathbb{P}_k[z_1, \ldots, z_\ell]$, then MPGMRES finds $\boldsymbol{r}^{(k)}$ such that

$$\left\|\boldsymbol{r}^{(k)}\right\| = \min_{\substack{p \in \mathbb{P}_k[z_1,\ldots,z_\ell] \\ p(0,\ldots,0)=1}} \left\|p(\mathcal{A}\mathcal{P}_1^{-1}, \ldots, \mathcal{A}\mathcal{P}_\ell^{-1})\boldsymbol{r}^{(0)}\right\|. \tag{8.10}$$

While the search space for complete MPGMRES is very rich, its exponential growth becomes prohibitive in practice. As such, a variant which selects only some of the potential search directions, ideally ensuring only linear growth, is natural to consider as a more practical alternative.

8.3 Selective MPGMRES (sMPGMRES)

To balance the benefits gained by adding multiple search directions with the storage and compute costs as we iterate, we might wish to fix the number of preconditioner applications and matrix–vector products independent of iteration. This would allow for parallelisation of these operations via use of a fixed number of processors. To do so, we consider limiting the growth of the search space in MPGMRES to be linear, with respect to the iteration number k, by using the *selective MPGMRES* (sMPGMRES) algorithm outlined in [88].

The search directions in MPGMRES are given in the collection of column vectors Z. To limit the growth of the search space we limit the size of Z, in particular here to be proportional to the number of preconditioners, independent of the iteration number. To do this we select only certain search directions from the span of the columns of Z, giving a selective MPGMRES algorithm. There are many strategies to selectively choose these directions, for instance instead of applying the preconditioners to all columns of $V^{(k)}$, as in (8.4), we might apply them to just a single vector from $V^{(k)}$, selecting this vector differently for each preconditioner. This selection choice need not be the same at each iteration and could incorporate randomness if desired. The corresponding $Z^{(k)}$ is then

$$Z^{(k)} = \left[\mathcal{P}_1^{-1} V_{:,s_1}^{(k)}, \ldots, \mathcal{P}_\ell^{-1} V_{:,s_\ell}^{(k)}\right], \tag{8.11}$$

where $V_{:,s_i}^{(k)}$ is the s_ith column of $V^{(k)}$ and s_i might change with k. One could further imagine defining s_i so as to include a small subset of the columns.

An alternative to applying each preconditioner to just one vector from $V^{(k)}$ (or a small subset of them) is to apply them all to a linear combination of these vectors, namely to $V^{(k)}\boldsymbol{\alpha}^{(k)}$ for some vector $\boldsymbol{\alpha}^{(k)}$ of appropriate size detailing the contribution from each column of $V^{(k)}$. The corresponding $Z^{(k)}$ is then

$$Z^{(k)} = \left[\mathcal{P}_1^{-1}V^{(k)}\boldsymbol{\alpha}^{(k)}, \ldots, \mathcal{P}_\ell^{-1}V^{(k)}\boldsymbol{\alpha}^{(k)}\right] \in \mathbb{R}^{n\times\ell}. \tag{8.12}$$

A natural choice for $\boldsymbol{\alpha}^{(k)}$ might be the vector $\mathbf{1}$, of all ones. All of these selection methods result in choosing some lower dimensional subspace of the full space and minimising, as in (8.10), over this subspace. With these selection strategies, where we limit $Z^{(k)}$ to ℓ new directions each iteration, then in our Arnoldi-type decomposition $\widetilde{V}_{k+1}$ has $k\ell + 1$ basis vectors while the number of columns of $\widetilde{Z}_k$ is $k\ell$. Hence, the storage is proportional to k, as in FGMRES, as opposed to exponential in k, like complete MPGMRES.

Now suppose we have reason to favour one preconditioner over another and, for simplicity, that there are just two candidate preconditioners $\mathcal{P}_1$ and $\mathcal{P}_2$. We would like our selective approach to incorporate knowledge of which preconditioner to favour. To do so we might choose an $\boldsymbol{\alpha}^{(k)} = \boldsymbol{\alpha}$ so that we weight more the contributions coming from one of the preconditioners. Consider the initial steps in sMPGMRES: we start with new search directions $Z^{(1)}$ and orthogonalise them to be $V^{(2)}$ as

$$Z^{(1)} = \beta^{-1}\left[\mathcal{P}_1^{-1}\boldsymbol{r}^{(0)}, \mathcal{P}_2^{-1}\boldsymbol{r}^{(0)}\right] \overset{\text{orthog.}}{\longrightarrow} V^{(2)}, \tag{8.13}$$

then add search directions $Z^{(2)}$ which are orthogonalised to be $V^{(3)}$

$$Z^{(2)} = \left[\mathcal{P}_1^{-1}V^{(2)}\boldsymbol{\alpha}, \mathcal{P}_2^{-1}V^{(2)}\boldsymbol{\alpha}\right] \overset{\text{orthog.}}{\longrightarrow} V^{(3)}. \tag{8.14}$$

So $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)^T$ weighs the contributions from each of the two preconditioners as $V^{(2)}\boldsymbol{\alpha} = \alpha_1 V^{(2)}_{:,1} + \alpha_2 V^{(2)}_{:,2}$ and the two columns of $V^{(2)}$ come from the two different preconditioned residuals. If we let $\boldsymbol{\alpha} = (\alpha, 1-\alpha)^T$, for some $\alpha \in (0,1)$, then the parameter α states how much we favour the first preconditioner, with $\alpha = \frac{1}{2}$ giving equal weighting and being equivalent to using the vectors of all ones as suggested above. Similar strategies could be used to weight contributions from more than two preconditioners.

Clearly in this weighted version of sMPGMRES the ordering of the preconditioners $\mathcal{P}_1, \ldots, \mathcal{P}_\ell$ is important as we weight them differently. However, even with equal weighting (using $\boldsymbol{\alpha} = \mathbf{1}$) ordering is important. This more nuanced asymmetry within sMPGMRES is an aspect not mentioned in [88]. The asymmetry comes about from

the need to orthogonalise the new search directions in $Z^{(k)}$ within themselves. The contribution from the first preconditioner is allowed to be in any new direction but this direction is taken out of the contribution from subsequent preconditioners, and so on as we orthogonalise *in order* the contributions from all the preconditioners. This means that if the direction from the last preconditioner is mostly within the span of the preceding directions it may well contribute little of value, despite coming from a good preconditioner when applied by itself. As a general rule then, we might value less these final search directions as the useful components may have already been taken out. This suggests taking a weighting $\boldsymbol{\alpha}$ which decreases in the components (instead of being equal) might be preferred. Nonetheless, in practice with a small number of good preconditioners, $\boldsymbol{\alpha} = \mathbf{1}$ might suffice to be as good. We will see that when we favour a preconditioner the ordering will matter, even if we are weighting the preconditioners in the same way. Further, ordering can still have a significant impact even when the preconditioners are weighted equally, especially when one of the preconditioners is poorer.

8.4 Numerical results for sMPGMRES

There are a number of questions to address in the performance of sMPGMRES. Can we match or improve on the performance of the best preconditioner? Is the performance robust against a preconditioner performing poorly? How sensitive is it on the weighting of the preconditioners?

In this section we apply sMPGMRES with the ideal two-phase PCD and LSC$(\mu, 1)$ (with $\epsilon^{-1} = 10$) methods. To answer questions of robustness we also use the SIMPLE-like preconditioner. We restrict our results to focus on the two preconditioner case ($\ell = 2$) using (8.12) with $\boldsymbol{\alpha}^{(k)} = (\alpha, 1-\alpha)^T$ for some $\alpha \in (0, 1)$. We again consider the cavity and step problems (Problems 1 and 2) in IFISS and use the MATLAB implementation[2] which accompanies [88]. We focus on considering iteration counts, as opposed to timings, since the implementation runs in serial and so does not take advantage of the parallelism inherent in sMPGMRES. Here we provide a sample of representative results to help answer the above questions; additional results can be found in Appendix A. Note that, when we tabulate our results using sMPGMRES, the iteration counts given in bold emphasise the best choice of weighting parameter α which provides the minimum number of iterations for a given pair of preconditioners. The preconditioner given on the left is used as the first preconditioner in sMPGMRES.

[2] `www.mathworks.com/matlabcentral/fileexchange/34562-multi-preconditioned-gmres`

Table 8.1: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the PCD and LSC preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2\times10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), $h = 1/64$, and varying Reynolds number Re and time-step Δt.

		Weighted sMPGMRES Preconditioner										
Δt	Re	PCD	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	LSC
$\frac{1}{10}$	10	15	12	11	11	11	11	12	12	12	14	**10**
	$10^{1.5}$	15	12	**11**	**11**	**11**	**11**	12	12	13	14	**11**
	100	14	12	**11**	12	12	12	13	13	14	15	13
	$10^{2.5}$	14	**12**	**12**	**12**	**12**	**12**	13	14	15	17	15
	1000	16	**13**	**13**	**13**	**13**	14	14	15	17	19	17
1	10	17	15	14	**13**	**13**	**13**	14	15	15	16	14
	$10^{1.5}$	19	16	14	14	14	**13**	14	14	15	15	14
	100	23	17	16	15	**14**	**14**	**14**	**14**	15	17	**14**
	$10^{2.5}$	25	18	16	15	15	**14**	15	15	16	17	15
	1000	28	21	19	18	**16**	18	18	18	19	19	20
10	10	18	15	**14**	**14**	**14**	**14**	15	15	16	17	**14**
	$10^{1.5}$	22	18	17	16	**15**	16	16	17	17	18	16
	100	28	22	21	20	19	**18**	19	20	20	20	19
	$10^{2.5}$	34	27	26	**24**	**24**	**24**	**24**	29	28	26	26
	1000	43	41	34	**31**	32	32	34	34	34	39	40

Table 8.1 display results for the time-dependent cavity problem using Newton linearisation when combining LSC and PCD. We see that the best iteration counts are seen towards the centre of the table, that is with a weighting parameter α closer to $\frac{1}{2}$, though we see some bias towards larger α as the asymmetry of ordering might suggest (this is also seen if we reverse the ordering, as observed in Table A.5). In this example almost any choice of α will provide some improvement over either of PCD or LSC individually while the best choice can allow convergence using up to 22% fewer iterations. Note that the choice $\alpha = \frac{1}{2}$ typically gives iterations counts close to optimum. Given that it is not clear we necessarily should do any better than the best preconditioner by itself, these results are quite promising and show that sMPGMRES can improve performance in terms of the number of iterations required. Furthermore, we see in this case that the performance is not particularly sensitive to α. To examine robustness we consider combining the PCD or LSC approaches with the SIMPLE-like preconditioner.

Table 8.2: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the LSC and SIMPLE-like preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), $h = 1/64$, and varying Reynolds number Re and time-step Δt.

		Weighted sMPGMRES Preconditioner										
Δt	Re	LSC	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	SIMPLE
$\frac{1}{10}$	10	**10**	11	11	12	14	16	21	27	33	37	124
	$10^{1.5}$	**11**	12	13	14	16	19	21	24	29	38	112
	100	**13**	**13**	15	16	19	22	24	28	37	45	93
	$10^{2.5}$	**15**	**15**	17	18	21	24	28	32	35	38	85
	1000	**17**	18	19	21	24	27	35	44	54	54	81
1	10	14	**13**	14	15	17	18	28	30	36	40	137
	$10^{1.5}$	14	**13**	14	15	17	19	25	32	35	43	144
	100	14	**13**	14	15	17	19	24	32	39	48	146
	$10^{2.5}$	15	**14**	15	17	19	22	25	36	45	55	144
	1000	20	**19**	20	23	26	31	36	60	112	142	139
10	10	**14**	**14**	**14**	16	17	20	25	32	34	41	138
	$10^{1.5}$	16	**15**	16	17	19	21	26	32	39	40	150
	100	19	**18**	**18**	19	21	23	28	35	45	47	165
	$10^{2.5}$	26	24	**23**	**23**	24	26	30	38	45	61	188
	1000	40	36	35	35	**34**	36	38	41	51	77	234

Table 8.2 combines the LSC and SIMPLE-like preconditioners for this problem. We see that primarily there is very little to be gained from including the SIMPLE-like approach with the best choice either being to simply use LSC or else a large α favouring LSC. However, if we change the ordering to have the SIMPLE-like approach first, as in Table 8.3, the picture looks slightly different, this time any $\alpha \leq \frac{1}{2}$ gives results comparable to LSC. This suggests that, while we do not gain much in the way of improved performance, the algorithm is still fairly robust to varying α so long as we do not favour the poorly performing preconditioner too strongly. This example provides a case where, with equal weighting ($\alpha = \frac{1}{2}$), the ordering of the preconditioners can quite substantially matter. Furthermore, it is by putting the worst preconditioner first (which by the asymmetry is subtly favoured) that we obtain the better results. While at first this may sound counter-intuitive, we can make sense of this observation by considering what the selection in sMPGMRES is doing. If the good preconditioner is used first then we take this contribution away from that of the

Table 8.3: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the SIMPLE-like and LSC preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), $h = 1/64$, and varying Reynolds number Re and time-step Δt.

		Weighted sMPGMRES Preconditioner										
Δt	Re	SIMPLE	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	LSC
$\frac{1}{10}$	10	124	29	18	15	13	12	11	11	**10**	**10**	**10**
	$10^{1.5}$	112	27	18	15	13	12	11	11	**10**	**10**	11
	100	93	25	20	17	15	14	13	12	12	**11**	13
	$10^{2.5}$	85	26	22	18	17	15	15	15	**14**	15	15
	1000	81	27	24	22	20	19	19	19	20	21	**17**
1	10	137	28	20	16	15	**13**	**13**	**13**	**13**	**13**	14
	$10^{1.5}$	144	29	20	17	15	14	**13**	**13**	14	**13**	14
	100	146	33	21	17	15	15	**14**	**14**	**14**	**14**	**14**
	$10^{2.5}$	144	30	23	18	17	16	15	**14**	**14**	15	15
	1000	139	36	30	25	22	21	20	**19**	20	22	20
10	10	138	31	22	18	16	**14**	**14**	**14**	**14**	**14**	**14**
	$10^{1.5}$	150	29	21	18	17	16	**15**	**15**	**15**	16	16
	100	165	34	24	21	19	**18**	**18**	19	19	19	19
	$10^{2.5}$	188	38	30	26	25	24	**23**	**23**	24	25	26
	1000	234	49	39	36	34	**33**	**33**	36	37	37	40

second preconditioner, likely making it even worse, then by equally weighting these we are allowing a large component of this much worse contribution to prevail. On the other hand, if the worse preconditioner is first, we remove this component from the contribution of the better preconditioner, which is unlikely to make this contribution worse and may possibly make it even better. Thus we see this latter combination is more favourable than the former, though we may not expect it to provide significantly better results than the best preconditioner by itself.

We see a similar scenario to the above in Tables A.6 and A.7, where the PCD and SIMPLE-like preconditioners are combined, with the SIMPLE-like approach mostly contributing little improvement over using just the PCD method. In this case the range of α giving reasonable performance is smaller than with LSC but again ordering the SIMPLE-like preconditioner first allows for greater robustness with respect to choosing α. It is also interesting to note that strong weighting towards the SIMPLE-like approach is still much better than only using the SIMPLE-like preconditioner.

Table 8.4: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the LSC and PCD preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady backwards-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h.

		Weighted sMPGMRES Preconditioner										
Re	h	LSC	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	PCD
10	$\frac{1}{4}$	22	17	**16**	**16**	18	17	19	20	21	26	26
	$\frac{1}{8}$	14	14	**13**	**13**	**13**	14	14	16	18	21	24
	$\frac{1}{16}$	14	14	**13**	**13**	**13**	**13**	15	15	19	20	24
	$\frac{1}{32}$	14	15	15	14	**13**	**13**	14	14	18	19	23
100	$\frac{1}{4}$	27	24	22	**20**	**20**	21	22	24	25	29	31
	$\frac{1}{8}$	24	20	19	19	**18**	**18**	20	22	24	28	30
	$\frac{1}{16}$	23	20	**19**	**19**	**19**	**19**	20	23	25	28	30
	$\frac{1}{32}$	23	20	**19**	**19**	**19**	20	22	24	28	30	31
500	$\frac{1}{4}$	77	69	65	62	57	56	**52**	59	74	85	60
	$\frac{1}{8}$	73	72	71	64	60	**51**	53	52	60	64	55
	$\frac{1}{16}$	65	67	63	49	49	46	**43**	**43**	52	47	49
	$\frac{1}{32}$	63	52	47	44	39	**36**	**36**	41	39	41	47

Table 8.4 provides results for combining LSC and PCD, this time for the steady step problem using Newton linearisation. As with the cavity problem, we again see that performance is not particularly sensitive to α and we observe similar behaviour for $\alpha \in [0.3, 0.7]$. When we now combine the PCD approach with the SIMPLE-like preconditioner for the steady step problem, as shown in Tables 8.5 and 8.6, we do obtain noticeably improved results, achieving up to a 38% reduction in the number of iterations required. A similar picture is observed when combining LSC with the SIMPLE-like preconditioner, this time yielding up to a 27% reduction; see Tables A.9 and A.10. Again, putting the poorer preconditioner first in the ordering typically yields the better choice and provides robustness, with α up to around 0.7 achieving similar or better performance than the best individual preconditioner. These results are somewhat promising and suggest that sMPGMRES can be used to help provide robustness when several alternative preconditioners are available.

Table 8.5: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the PCD and SIMPLE-like preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady backwards-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h.

		Weighted sMPGMRES Preconditioner										
Re	h	PCD	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	SIMPLE
10	$\frac{1}{4}$	26	**16**	**16**	17	**16**	17	20	22	33	39	38
	$\frac{1}{8}$	24	16	**15**	16	17	18	20	26	34	50	58
	$\frac{1}{16}$	24	**15**	16	16	17	19	21	27	37	56	89
	$\frac{1}{32}$	23	**16**	**16**	17	17	20	24	33	55	62	137
100	$\frac{1}{4}$	31	20	**19**	20	20	21	25	36	43	49	55
	$\frac{1}{8}$	30	**19**	**19**	**19**	**19**	**19**	21	24	40	60	82
	$\frac{1}{16}$	30	20	20	**19**	**19**	**19**	22	28	37	69	121
	$\frac{1}{32}$	31	22	21	**20**	**20**	**20**	23	32	43	59	179
500	$\frac{1}{4}$	60	58	**56**	**56**	64	79	95	105	105	106	115
	$\frac{1}{8}$	55	**52**	54	56	59	74	113	216	261	253	190
	$\frac{1}{16}$	49	48	41	**40**	41	45	50	62	99	156	207
	$\frac{1}{32}$	47	37	36	**35**	**35**	36	39	43	60	83	284

It remains to be seen how much performance improvement can be gained using multipreconditioning in terms of computation time using a parallel implementation. However, results in [88] on a single-phase driven cavity problem using PCD and LSC (without boundary modifications) are promising and show that the fastest algorithm for most problem instances, in particular for finer meshes, is sMPGMRES. Our results show a similarly positive outlook and further that weighting can be introduced to favour a particular preconditioner, in which case the algorithm is not particularly sensitive to any sensible choice of weight. Furthermore, when incorporating a poorer preconditioner into the multipreconditioning approach, the algorithm can still remain reasonably robust against such a poorer method (especially if this method is ordered first). This suggests that sMPGMRES may help provide robustness in simulations of two-phase flow where different preconditioners may work better in different regimes, though this claim remains to be verified.

Table 8.6: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the SIMPLE-like and PCD preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady backwards-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h.

		Weighted sMPGMRES Preconditioner										
Re	h	SIMPLE	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	PCD
10	$\frac{1}{4}$	38	27	20	18	**17**	**17**	**17**	**17**	**17**	**17**	26
	$\frac{1}{8}$	58	33	23	19	18	17	**16**	18	17	19	24
	$\frac{1}{16}$	89	42	27	21	18	**17**	**17**	**17**	**17**	**17**	24
	$\frac{1}{32}$	137	53	28	23	19	18	18	**16**	17	18	23
100	$\frac{1}{4}$	55	49	34	24	21	**19**	**19**	20	21	21	31
	$\frac{1}{8}$	82	52	32	22	20	**19**	**19**	**19**	20	21	30
	$\frac{1}{16}$	121	60	32	23	20	**19**	20	**19**	21	22	30
	$\frac{1}{32}$	179	69	36	25	22	**20**	21	21	22	22	31
500	$\frac{1}{4}$	115	103	88	69	60	**53**	54	56	61	62	60
	$\frac{1}{8}$	190	209	131	78	62	56	**50**	52	53	54	55
	$\frac{1}{16}$	207	127	77	50	41	**39**	**39**	40	42	42	49
	$\frac{1}{32}$	284	98	52	40	37	**35**	**35**	36	37	38	47

Finally, we remark that such a multipreconditioning approach might also provide an alternative to adaptively changing the preconditioner by trying to monitor the flow regime and using previously known results about when each preconditioner performs best. Further work is required to investigate these ideas.

Chapter 9

Towards industrial applications

In this chapter we detail work focused towards industrial applications, in particular utilising the Proteus computational methods and simulation toolkit being developed by CHL for solving large air–water free-surface problems that arise in coastal and hydraulic applications; see `http://proteustoolkit.org` for more details. We will discuss some of the additional challenges faced compared with the simpler formulation we have considered in the bulk of this work. We first detail the model used in Proteus before discussing how this affects our solver methodology. We then present numerical results for a fully dynamic test problem.

Note that work in this chapter is joint with another student, Alistair Bentley, who has implemented the majority of our solver methodology in Proteus.

9.1 The model used in Proteus

The implementation for the two-phase Navier–Stokes flow model in Proteus is found in the Reynolds Averaged Two Phase Navier–Stokes (RANS2P) module of the toolkit. In depth details are provided in the technical report [13]. We begin by discussing the conservative level set formulation implemented within RANS2P before addressing the form of Navier–Stokes equations and the stabilisation techniques used.

To detail the level set method, we first recall the basic level set equation

$$\frac{\partial \phi}{\partial t} + \boldsymbol{u} \cdot \nabla \phi = 0, \tag{9.1}$$

for a level set function ϕ. This function distinguishes the two fluids by making use of the Heaviside function $H(\phi)$ when defining the material parameters (namely, density and viscosity). The fluid velocity $\boldsymbol{u}$ is calculated using a set of Navier–Stokes equations and used to update the level set function by solving a discretisation of (9.1). This allows us to advance the interface $\Gamma(t)$ in time, defined as the zero level set of ϕ.

However, the formulation resulting from using the basic level set equation (even with advective stabilisation; see (3.22)) typically results in mass conservation errors [120]. Instead, the conservative approach described in [120] is used and complements the basic level set equation with three auxiliary equations. The first of these is a re-distancing step for the level set function which maintains the zero level set defining the interface. An eikonal equation is solved to calculate a signed distance function ϕ_d, namely

$$\|\nabla \phi_d\| = 1 \quad \text{a.e. in } \Omega, \qquad\qquad \phi_d = 0 \quad \text{on } \Gamma. \tag{9.2}$$

Next we determine a mass-conserving approximation of the pointwise volume fraction for the first fluid, denoted $\widehat{H}$. Note that if we could solve the level set equation (2.2) exactly then the pointwise volume fraction would simply be $H(\phi)$. The underlying conservation of fluid mass equation we wish to emulate is

$$\frac{\partial \left(\rho H(\phi)\right)}{\partial t} + \nabla \cdot (\rho H(\phi) \boldsymbol{u}) = 0. \tag{9.3}$$

After eliminating density using the incompressibility constraint, [120] solves for the mass conserving approximation $\widehat{H}$ using

$$\frac{\partial \widehat{H}}{\partial t} + \nabla \cdot (\widehat{H} \boldsymbol{u}) = 0. \tag{9.4}$$

Finally, the volume fraction and signed distance function are linked through a non-linear reaction–diffusion equation to give a correction ϕ' to ϕ_d given by

$$\zeta(\nabla \cdot \nabla)\phi' = H(\phi_d + \phi') - \widehat{H} \quad \text{in } \Omega, \qquad \nabla \phi' \cdot \boldsymbol{n} = 0 \quad \text{on } \partial\Omega, \tag{9.5}$$

where ζ is a parameter that penalises the derivation of ϕ' from being a global constant. It is noted in [120] that these latter two equations apply to any two-phase system with an underlying mass or volume balance principle for the phases. An illustration of how these auxiliary equations are solved in practice is found in [120].

We now turn our attention to the form of Navier–Stokes equations used. RANS2P allows for both conservative and non-conservative forms of the equations [13]; here we detail the non-conservative form akin to that considered in Chapter 2. We note that the underlying Navier–Stokes equations remain the same, being given by

$$\rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho\, \boldsymbol{u} \cdot \nabla\, \boldsymbol{u} - \nabla \cdot (2\mu\, \boldsymbol{D}\boldsymbol{u}) + \nabla\, p = \rho\, \boldsymbol{g} \qquad \text{in } \Omega, \tag{9.6a}$$

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{in } \Omega, \tag{9.6b}$$

where we now specify the forcing term $\boldsymbol{g}$ to be acceleration due to gravity. However, the weak form is structured differently from that introduced in Chapter 3, in particular for the pressure and incompressibility constraint, as we now detail.

The weak formulation of the momentum equation (9.6a) is derived similarly to that in (3.10), however we now do not apply integration by parts on the pressure term (thus assuming the weak gradient of the pressure exists). Namely we have

$$\int_\Omega \rho \left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} \right) \cdot \boldsymbol{v} + \int_\Omega 2\mu \operatorname{Tr}(\boldsymbol{D}\boldsymbol{u}\, \boldsymbol{D}\boldsymbol{v}) + \int_\Omega \nabla p \cdot \boldsymbol{v} = \int_\Omega \rho \boldsymbol{g} \cdot \boldsymbol{v}, \tag{9.7}$$

where now we note that the arising boundary term is just

$$- \int_{\partial\Omega} 2\mu\, \boldsymbol{v} \cdot \boldsymbol{D}\boldsymbol{u} \cdot \boldsymbol{n}, \tag{9.8}$$

as such, different naturally enforced boundary conditions arise involving this flux term rather than the full stress tensor. For the incompressibility constraint we now apply integration by parts to give a similar coupling term between the pressure and velocity variables to that in the momentum equation (9.7), namely we have

$$\int_\Omega \nabla q \cdot \boldsymbol{u} = - \int_{\partial\Omega} \boldsymbol{u} \cdot \boldsymbol{n}\, q. \tag{9.9}$$

A benefit of this formulation is that it allows easier access to relevant flux quantities (see e.g. [12, 235]) which are important in fluid–structure interaction problems.

For discretisation, equal order continuous (Galerkin) finite elements are used on simplex meshes, in particular we consider stabilised $\boldsymbol{P}_1$–$\boldsymbol{P}_1$ elements (continuous piecewise linear approximations for both velocity and pressure on the same mesh of elements). Since simulations of interest typically run at high Reynolds number, being highly advection dominated, advective stabilisation is required as well as a stabilised form of $\boldsymbol{P}_1$–$\boldsymbol{P}_1$ discretisation. RANS2P uses a variation of algebraic subgrid scale (ASGS) stabilisation to cover both of these issues as well as a discontinuity capturing term which amounts to the addition of numerical diffusion; see [13]. Further background on such stabilisation techniques can be found in [223, 109, 50]. We omit a detailed description of the formulation but note that the resulting stabilisation adds additional terms into the discretised problem and that such terms may affect the efficacy of any preconditioning techniques that are utilised.

As noted above, due to the weak form used, the boundary conditions that can be enforced naturally are different from those arising in the standard weak formulation of the Navier–Stokes equation. However, simulations in RANS2P often incorporate weakly enforced (Nitsche-type) boundary conditions [12, 5]. Again, this requires the addition of further terms, this time boundary integral terms, into the formulation. In our results we will study an enclosed flow problem which allows for strongly enforced boundary conditions to be easily implemented, and so we use this approach.

9.2 The solution methodology used in Proteus

We now briefly outline the solver methodology we make use of. For time-stepping, Proteus has a variety of backwards differentiation formulae (BDF) available, however, we stick to the simplest, namely the backwards Euler method in line with our previous computations. To solve the resulting discrete nonlinear system Newton iteration is employed (see Section 3.4.2) with the initial guess being the solution calculated at the previous time-step, except for the very first time-step in which case it is taken to be the zero vector. So long as the time-steps are not too large, this allows for the Newton iteration to converge.

As before, this nonlinear iteration yields a sequence of linear systems to be solved, each having the (generalised) saddle point form (3.36) but now with the additional terms discussed above included. The RANS2P module makes use of PETSc and PETSc for Python to solve these linear systems [11, 55]. Typically an iterative solver such as GMRES is used to solve the linear systems. We again focus on pairing GMRES with Schur complement based preconditioning techniques, which can be implemented using the 'fieldsplit' functionality in PETSc.

For preconditioning, the upper block triangular form of preconditioner (4.11) is employed and we utilise the (split) two-phase PCD (5.64) and SIMPLE-like (4.18) inverse Schur complement approximations. We note that the SIMPLE-like approach to approximating the inverse Schur complement is implemented in PETSc under the name 'selfp'. Such an approach can perform well if the time-steps taken are small enough so that $\mathrm{diag}(F)$ well approximates F in (3.36). We remark that in this chapter we will not consider the LSC approach for several reasons. The main reason for this is that RANS2P typically uses simplex meshes and, on such meshes, it has previously been observed that the LSC method displays poor scaling performance upon mesh refinement [152]. Moreover, since RANS2P primarily uses equal order elements for the velocity and pressure approximation spaces. This would require modifying the LSC preconditioner to account for the necessary finite element stabilisation, similarly to Chapter 6. However, the ASGS stabilisation and variable coefficient nature of the problem further complicates proper handling of such stabilisation terms.

We now discuss these Schur complement approaches in the context of the Proteus model previously outlined. Firstly, since the discrete formulation requires a variety of additional terms, the operator that F represents in (3.36) is more complicated than the one used to derive the PCD approach; clearly the approximation in (5.64) does not capture these additional terms. However, one term which can be easily taken into

account is the numerical viscosity from the discontinuity capturing term. Here we can simply add the numerical viscosity to the physical viscosity μ in the relevant mass matrix in (5.64). It remains to be seen if further appropriate terms within the inverse Schur complement approximation can be incorporated to improve performance of the PCD approach for the Proteus formulation. On the other hand, since the SIMPLE-like approach is purely algebraic, it captures the additional terms in the linear system and no modifications are required. However, how well it can capture such terms is still dependent on how well diag(F) approximates the important features of F.

Before moving forward, we say a few words about applying $\widehat{F}^{-1}$ (recall (4.11)). For large problems, it is important that the action of $\widehat{F}^{-1}$ is scalable and provides a reliable approximation to F^{-1}. Thus, developing effective methods to approximate F^{-1} is an active area of Proteus research. For advective dominated flows, however, finding an effective and robust $\widehat{F}^{-1}$ can be difficult [163]. Further complicating matters is the ASGS stabilisation used and the dynamic two-phase nature of the simulations. We have implemented an approach which treats $\widehat{F}^{-1}$ as a block preconditioned GMRES iteration where the sub-blocks are approximated using multigrid methods and this has shown signs of being effective both in our work and others [100]; see also results in [7]. Unfortunately, maintaining solver robustness throughout an entire multi-physics simulation remains an issue. As such, we defer discussion of this important topic to future work and focus our efforts here on the Schur complement approximation, taking $\widehat{F}^{-1}$ to be given by the sparse direct solver `SuperLU_DIST` [135, 134].

9.3 Numerical results for a dynamic dam-break problem

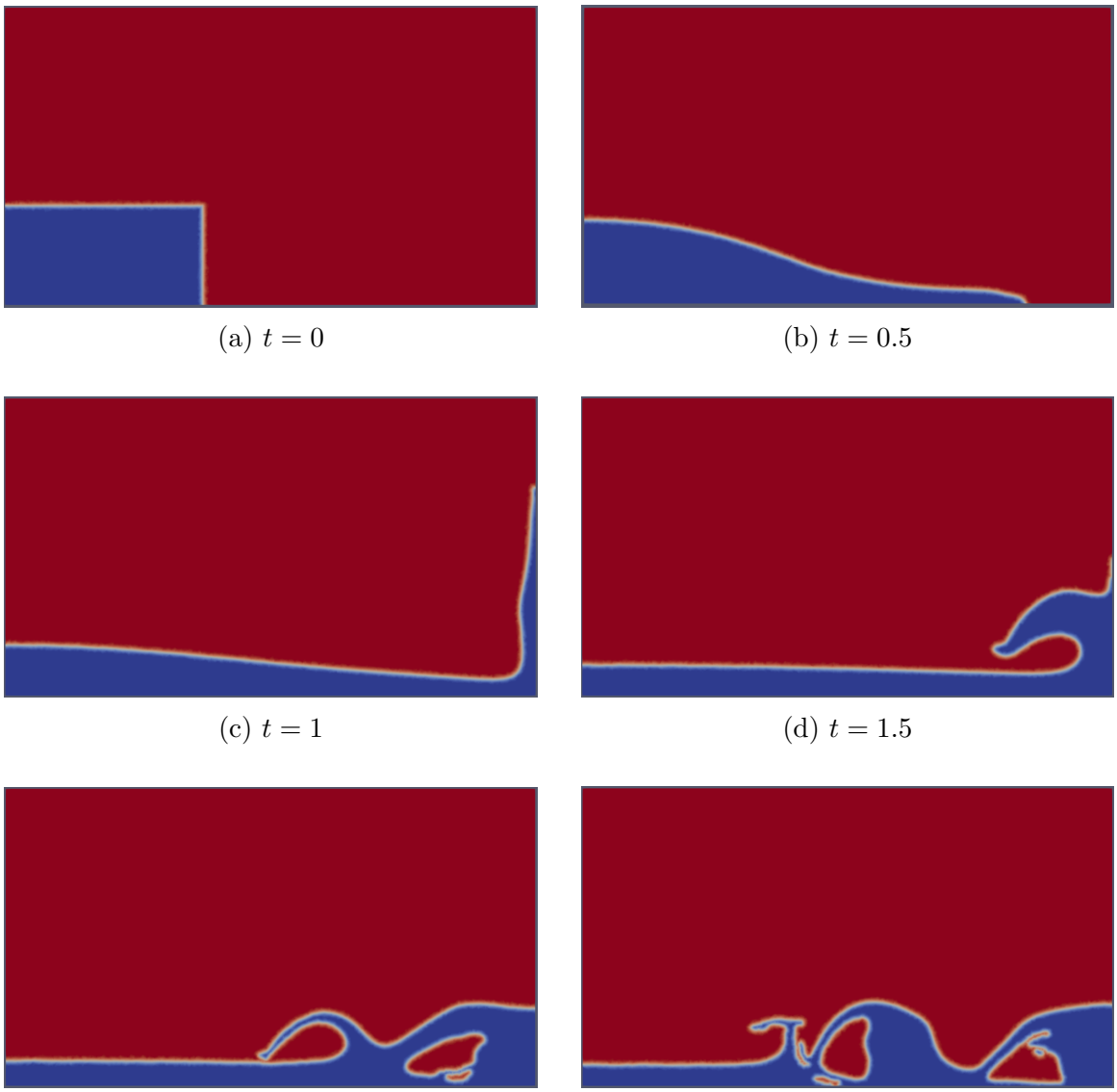
Free-surface models that accurately describe complicated air–water flow dynamics are an important application of the two-phase Navier–Stokes equations. For example, level set and volume-of-fluid methods can be combined with two-phase Navier–Stokes equations to simulate intricate hydraulic processes such as waves crashing into coastal structures. For these multi-physics models, the Navier–Stokes equations are just one component in a larger system of equations. Nonetheless, in almost all cases, it is solving the discrete linearised Navier–Stokes equations that is the most time consuming part of a multi-physics simulation. Thus, superior preconditioners have the potential to dramatically improve computational run times. In this section, we apply the two-phase PCD preconditioner in a free-surface model to gauge its effectiveness for dynamic multi-physics problems and compare it with the SIMPLE-like approach.

We consider a 2D benchmark dam-break problem of air–water flow described in [51, 254]. The domain is rectangular, with $\Omega = (0, 3.22) \times (0, 1.8)$, and the free-slip condition (2.11c) is applied everywhere on the boundary $\partial\Omega$. Initially, there is a standing column of water in $\Omega_1 = (0, 1.2) \times (0, 0.6)$ with the remaining space being occupied by the air. The simulation runs for a two second time interval and begins as the column of water collapses under gravity and proceeds to collide with the right-hand wall of the tank. This collision creates a wave and ultimately topological changes in the phases. Figure 9.1 displays several snapshots of the simulation. The dam-break problem provides a good benchmark for testing the two-phase PCD and SIMPLE-like preconditioners because its features are typical of many dynamic, multi-physics problems of practical interest.

To compare the scaling performance of two-phase PCD and SIMPLE, we consider two simulations. In the first, time-steps are selected to ensure that the CFL number is less than or equal to 0.9. Such restrictions are often necessary for nonlinear solver convergence and solution accuracy. However, in some important cases this restriction on the CFL number is not strictly necessary; see Remark 8. Thus, in the second simulation, we use a fixed time-step of $\Delta t = 0.01$. In this case, the CFL number is larger than one for much of the simulation, reaching a maximum of 20.5 and typically being above 2.5. Nonetheless, the time-step is still small enough to achieve nonlinear solver convergence and solution accuracy. Here, we set an absolute tolerance of 10^{-6} for the nonlinear residual and a relative tolerance of 10^{-6} for the linear residual. For both simulations, we analyse the average and maximum number of GMRES iterations required at six different levels of mesh refinement. These mesh refinement levels are selected so that by the final refinement, the physics of the simulation is sufficiently resolved to perform relevant engineering analysis.[1]

Remark 8 (Beyond the CFL limit) *Accurate computation of relevant quantities of interest, such as drag force, for fixed hydraulic structures or vessels, frequently results in quasi-steady flows. In particular, the free surface may tend toward a steady wake structure or standing wave pattern, and this structure dominates the force on the given structure. In these cases, it is frequently desirable to use a fixed time-step that results in CFL numbers significantly larger than one. Time-stepping is then carried out until the quasi-steady hydrodynamic conditions are reached or the quantity of interest has reached a constant or steady periodic value.*

[1]Note that for triangular elements we measure the mesh refinement using $h = \sqrt{2T_A}$, where T_A is the area of the largest triangle. For our test problem, the number of elements when $h = 0.025$ is 29328 while for $h = 0.0125$ it is 117353.



(a) $t = 0$ (b) $t = 0.5$

(c) $t = 1$ (d) $t = 1.5$

(e) $t = 1.75$ (f) $t = 2$

Figure 9.1: Evolution of the dam-break simulation in Proteus at selected points in times. The VOF (volume-of-fluid) is plotted with blue representing the water phase and red being the air.

Table 9.1 presents the average and maximum number of GMRES iterations taken during the first simulation with a restricted CFL number. These results suggest that, on average, the two-phase PCD and SIMPLE-like preconditioners both scale well with the mesh size. However, Table 9.1 also reveals that the maximum number of iterations for the two-phase PCD method increases as the mesh is refined. As seen in Figure 9.2, the performance of the PCD approach becomes less consistent as the phases begin to undergo topological changes around 1.5 seconds into the simulation. Specifically, we

Table 9.1: The average (maximum) number of GMRES iterations required across different meshes when running the dam-break problem ensuring that the CFL number is restricted to be less than or equal to 0.9 using the two-phase PCD and SIMPLE-like preconditioners.

	$h = 0.4$	$h = 0.2$	$h = 0.1$	$h = 0.05$	$h = 0.025$	$h = 0.0125$
Two-phase PCD	8 (13)	7 (11)	8 (14)	9 (18)	9 (20)	10 (28)
SIMPLE-like	8 (10)	8 (11)	9 (11)	9 (12)	10 (14)	11 (16)

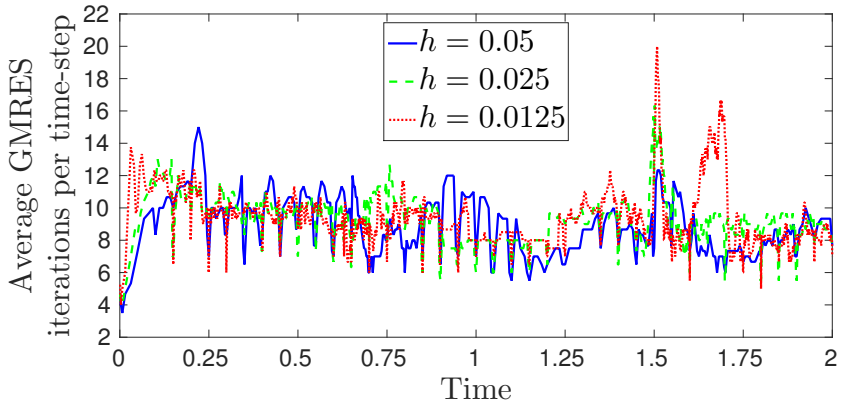
Table 9.2: The average (maximum) number of GMRES iterations required across different meshes when running the dam-break problem with fixed $\Delta t = 0.01$ using the two-phase PCD and SIMPLE-like preconditioners. Note that $\times$ indicates the simulation stopped due to failure in the solver convergence.

	$h = 0.4$	$h = 0.2$	$h = 0.1$	$h = 0.05$	$h = 0.025$	$h = 0.0125$
Two-Phase PCD	8 (13)	7 (11)	8 (14)	9 (18)	11 (21)	14 (27)
SIMPLE-like	8 (11)	8 (11)	8 (12)	9 (15)	19 (48)	$\times$

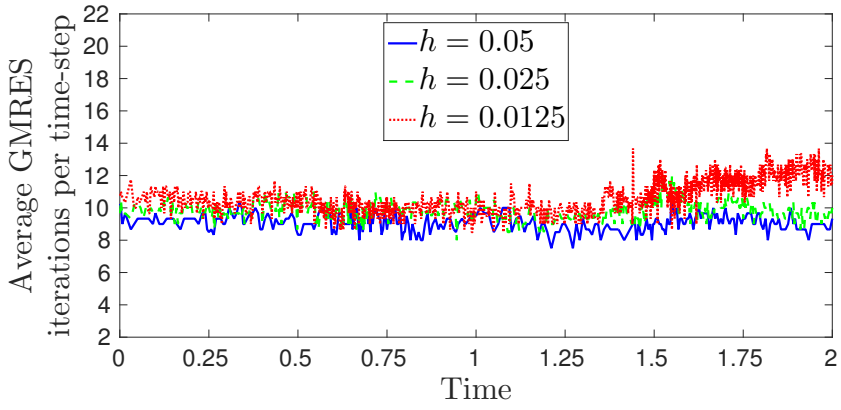
observe a sharp rise in iteration counts on all refinement levels just before the water splashing off the back wall reconnects with the fluid on the tank floor. This jump in iterations is likely a consequence of the dynamics within the continuous level set function as the water phase reconnects with itself. It is likely that the additional stabilisation terms which the PCD approach neglects become more prominent during this part of the simulation. Following on from this part, the two-phase PCD iteration begins to gradually show signs of scaling again (see Figure 9.2).

In contrast to the two-phase PCD preconditioner, the performance of the SIMPLE-like preconditioner remains relatively stable throughout the mixing processes. Indeed, this result is consistent with the algebraic nature of the SIMPLE-like approximation. That said, as the fluid phases mix, the SIMPLE-like preconditioner does exhibit some mesh dependence. This suggests that complicated topological phase changes pose a challenge for both preconditioners. Overall, while these results are encouraging for the two-phase PCD preconditioner, the method does not manage to match the consistency of the SIMPLE-like method or require significantly fewer iterations on average.

Results for the second simulation, using a fixed time-step $\Delta t = 0.01$, are shown in Table 9.2 and suggest that on coarse meshes, the two-phase PCD and SIMPLE-like preconditioners are competitive methods to one another. In contrast to the first simulation, however, as the mesh is refined, the performance of the SIMPLE-like method rapidly deteriorates while results for the two-phase PCD preconditioner remains stable, experiencing only a gradual increase in iterations being required.



(a) Two-phase PCD preconditioner



(b) SIMPLE-like preconditioner

Figure 9.2: Average preconditioned GMRES iterations per time-step for the two-phase PCD and SIMPLE-like preconditioners when running the dam-break problem ensuring that the CFL number is restricted to be less than or equal to 0.9.

These results are consistent with the steady state performance observed above in Chapter 7. As the mesh is refined for a fixed time-step, the advective features of the system become more pronounced and the CFL number increases. As we previously observed, the SIMPLE-like approach does not capture the features of an advection dominated flow well enough to provide a robust preconditioner. The two-phase PCD preconditioner, however, does account for such features and thus remains capable of producing stable, reliable results in this setting. Nonetheless, the average iteration counts required by two-phase PCD slowly increase as the mesh is refined, suggesting that inverse Schur complement approximation does not capture all of the important features of the true Schur complement within the more complicated Proteus model.

Overall, these preliminary results for the two-phase PCD preconditioner in a free-surface, multi-physics setting are encouraging. When a restricted CFL number is used, the two-phase PCD operator performs in line with the SIMPLE-like method, but does reveal some inconsistency during periods of rapid topological change. As the CFL number of the flow increases, however, the two-phase PCD operator begins to provide a significant improvement to the SIMPLE-like approach owed due to its ability to better capture the advective features of the flow. This suggests the PCD methodology may be advantageous in circumstances where large time-steps are to be taken, such as when quasi-steady flow solutions are required to be computed.

Nonetheless, many questions remain in order to address the suitability of the PCD approach in real applications. Such questions include the aforementioned issue of incorporating suitable corrections to the inverse Schur complement approximation to address the additional stabilisation terms featuring in the model. Furthermore, we have not discussed wall clock time and the overheads required in constructing the approach PCD operators. Such considerations are imperative for practitioners. It further remains to be seen how performance varies in large-scale simulations and when parallelisation is essential. However, in these first steps we see potential for such a specialised method to provide significant improvements in certain two-phase flow simulations.

Chapter 10

Conclusions and further work

10.1 Summary and conclusions

In this thesis we have considered preconditioning techniques for the linear systems that arise in simulations of incompressible two-phase flow in order to help towards developing scalable solution methodology. Our focus has primarily been on developing and testing extensions of the pressure convection–diffusion (PCD) and least-squares commutator (LSC) preconditioners for two-phase flow.

In Chapter 5 we used the viewpoint that the PCD approach generalises the Cahouet–Chabard preconditioner from Stokes to Navier–Stokes flow which allowed us to propose an extension to the PCD approach which incorporated appropriate scaling to represent the two-phase nature of the problem. However, the best choice of scaling remained unclear and the arguments involved did not include consideration of suitable boundary conditions within the pressure space operators that define the preconditioner. To remedy these issues we introduced the novel concept of weighted commutators. Examining such commutators allowed for a greater understanding of scaling within the PCD approach as well as on the issue of boundary conditions. Our study led us to derive an appropriate bi-weighted commutator which was able to fully specify what is required in order to build a two-phase PCD preconditioner.

As well as extending the PCD preconditioner to variable coefficient Navier–Stokes flow, we emphasised a viewpoint that the PCD approach is best considered in terms of continuous operators with the resulting inverse Schur complement approximation being the discretisation of an appropriate operator. This viewpoint is subtly different from those given in the literature and removes any reference of a discrete commutator that is to be minimised. Since the continuous operators come equipped with boundary conditions, these feature naturally in the development of the approach. Furthermore, this viewpoint proves particularly helpful in the two-phase scenario.

Our work also showed how the required Robin condition for the operator $\mathcal{F}_p$ can be implemented in a novel way when using continuous finite elements through the use of integration by parts in the weak formulation. Our new implementation was able to provide up to a 20% reduction in the number of iterations required for a model test problem. We also considered a splitting of the $\mathcal{F}_p$ operator and saw that in certain cases, due to the practical implementation involved, up to 50% more iterations can be required when not utilising a split form of PCD.

Overall, use of the two-phase PCD preconditioner was able to provide a scalable solver for the linear systems encountered in our simplified test problems implemented in IFISS with results showing effectively mesh-independent convergence in all cases.

In Chapter 6 we turned our attention to the LSC approach. Several LSC-type methods have already been introduced for the variable viscosity Stokes problem, however, we saw that performance tended to be poorer in the case of two-phase flow when the variable viscosity is discontinuous or else has large gradients if the interface is smoothed over a narrow transition region. In particular, for previously proposed methods, mesh-independence was lost in our results unless the viscosity was smooth and fixed independently of the grid size. On the other hand, we proposed a scaling within the LSC approach the which demonstrated mesh-independence and could significantly reduce the iteration counts required for a two-phase test problem with large viscosity contrasts compared with the best existing method. Our choice of scaling provided results which were close to being independent of the viscosity ratio, though care may be needed when using large viscosity contrasts to ensure any smoothing of the interface is sufficiently resolved by the mesh.

Following our investigation of the Stokes problem, we then turned to the more challenging case of Navier–Stokes flow. Here, there are two variable coefficients, namely the density and viscosity, and a universally appropriate choice of scaling in the LSC approach was less clear. Furthermore, unlike with the PCD approach, a choice of norm must be made. This open choice leaves some scope for different methods to be considered and so we explored this issue numerically, testing several different choices which yield different scaling in the resulting LSC preconditioner. Nonetheless, we showed how scaling incorporated in the LSC approach can be separated into two parts: one can be viewed as stemming from appropriate weighted commutators while the other resides in the choice of norm. Our interpretation of an underlying weighted commutator allowed us to suggest why our proposed LSC-type method in the Stokes case was superior for large viscosity ratios.

However, one potential issue in the LSC approach is the imposition of boundary adjustments, which are important in order to gain scalable performance. We saw that scaling in the choice of norm could interfere with these boundary adjustments in a negative way so that some mesh-dependence is seen. If we only scale the norm to include boundary adjustments then nearly mesh-independent behaviour is observed, providing a suitable choice of damping parameter is used, but higher iteration counts are seen in some instances, in particular for high Reynolds numbers flows.

A second issue we observed for LSC preconditioning applied to two-phase flow was that the multigrid technique we utilised for a more practical implementation tended to struggle somewhat with the scaled Laplacian sub-problems which featured. Performance was more erratic and higher iteration counts were observed than in the ideal implementation. Most prominently, mesh-dependence was seen to be poorer in the AMG iterated approach that we used. This highlighted a lack of robustness in approximating the solution to these sub-problems for the LSC approach which was not apparent for the discretisation choices made in our implementation of a two-phase PCD preconditioner. Nevertheless, the practical implementation can still provide reasonably good results. Our experience suggests that, while the LSC preconditioner can be very effective, overall the approach is less favourable compared to PCD in the two-phase case due to a typically unclear choice of optimal damping parameter and some difficulty in obtaining a scalable practical implementation. Further investigation into these topics may help shed light onto these issues and allow for more reliable and scalable performance to be obtained.

Nonetheless, Chapter 7 demonstrated that our new variants of the PCD and LSC preconditioners significantly improve performance for two-phase flow and are favourable compared to some of the existing approaches available in the literature. We highlight the fact that our two-phase PCD preconditioning approach displays excellent performance without any parameters to be tuned.

In Chapter 8 we shifted our focus from preconditioning to the underlying iterative method. We studied a multipreconditioning approach which allows for two or more preconditioners to be incorporated, whose application can be carried out in parallel. In practice we studied the case of utilising two preconditioners. We incorporated a modification which allows us to weight the contribution of each preconditioner and so favour one over the other. We further identified a source of asymmetry in the approach (even with the two preconditioners equally weighted) which had not been discussed before. This means that the ordering of the preconditioners used in the multipreconditioned technique can be important and we see this in our results.

Overall, we found that multipreconditioning can be successfully used to combine preconditioners and improve performance. Furthermore, we found that incorporating a worse preconditioner need not cripple the overall performance and can sometimes still reduce the number of iterations required. It is interesting to note that this feature suggests that multipreconditioning can be used not only to improve performance but to provide robustness. For example, it may be that, over the course of a simulation which features disparate flow regimes, different preconditioners may be preferable at different points within a simulation with these points being unknown in advance. Multipreconditioning offers the opportunity to use several different preconditioners simultaneously without needing to track the current flow regime so as to aim to switch preconditioner accordingly. Further work is needed in order to see if such improved robustness can be achieved in practice for two-phase flow simulations.

Finally, in Chapter 9 we looked towards industrial applications and, in particular, considered the model used within the Proteus software being developed at CHL. We outlined some of the additional challenges that are faced for preconditioning within these models. Numerical results were presented for a dynamic dam-break problem in Proteus which compared a variant of our two-phase PCD approach with a SIMPLE-like block preconditioner. The latter approach is straightforward to implement and can work well providing the time-steps used are sufficiently small. We saw that for the case of the time-steps being restricted by a CFL condition, similar iteration counts are seen for both the PCD and SIMPLE-like approaches. However, when larger time-steps are permitted we observed that PCD became preferable and was more robust than the SIMPLE-like preconditioner, which failed to converge on the finest mesh used. This suggests that the PCD methodology may be advantageous in circumstances where large time-steps are to be taken, such as when quasi-steady flow solutions need computing.

Nonetheless, several outstanding matters remain in order to better incorporate some of the features of the Proteus two-phase flow model into the PCD preconditioner. In particular, we saw some degradation in performance of PCD as the fluids mixed and underwent topological changes. This is likely due to the stabilisation terms that Proteus uses (which are not accounted for in the PCD approach) becoming more prominent at such points in the simulation. In general, the more additional features of the underlying model which can be incorporated into the PCD approximation, the better the resulting preconditioner will be. While we have laid the foundations, much is left to explore in order to further improve performance when simulating real problems of interest.

10.2 Further directions of study

While our work has shown some promise of utilising block preconditioning techniques which derive from approximate commutators for simulations of two-phase flow, there is much that remains if such approaches are to become well used for industrial scale problems accurately modelling hydraulic processes.

A key component of the block preconditioners we have discussed is an effective approximation $\widehat{F}$ to the variable coefficient convection–diffusion operator represented in the F block of the saddle point system (4.1). In the single-phase case, effective multigrid techniques can be used. However, our experience suggests that finding effective and robust multigrid methods is more challenging for the two-phase problem. The primary difficulty resides in the highly varying coefficients and the fact that the viscous operator no longer simplifies to a vector Laplacian term and contains non-zero off-diagonal sub-blocks. In order to assess the raw potential of the preconditioners we introduced we have used a direct solver to give the action of $\widehat{F}^{-1}$ throughout. However, this approach will not yield an efficient preconditioner for large-scale problems. It is therefore essential to further investigate the use of more robust multigrid methods, or other techniques such as domain decomposition, to provide the action of $\widehat{F}^{-1}$. Some work on methods for highly varying coefficients can be found in [84, 85, 199, 200].

Within the Proteus formulation we have implemented an approach which treats $\widehat{F}^{-1}$ as a block preconditioned GMRES iteration where the individual sub-blocks are approximated using multigrid methods, rather than the full F block. This has shown signs of being a promising approach both in our work and others such as [100], where a similar approximation is required for the augmented Lagrangian preconditioner; see also results in [7]. However, in our case there were still instances where this approach exhibited a lack of robustness and failure of the solver. Nonetheless, this approach is worthy of further investigation. We anticipate that appropriate and robust multigrid (or domain decomposition) methodology can be constructed but this may require operator-dependent grid transfers, more specialised smoothers, and further tuning compared to standard off-the-shelf methods such as those we have used.

We also saw that the AMG method we applied to approximate the scaled Laplacian sub-problems arising in the LSC preconditioner can sometimes show degradation in its performance. Again, consideration of other multigrid methods should be carried out along with tuning parameters, such as the number of V-cycles or type of multigrid cycle used. This will likely improve performance of a practical implementation of the LSC-type preconditioners considered and yield a more scalable approach.

Another area which we have not addressed in this work is an analysis of the new preconditioning techniques derived. We recall that eigenvalue bounds have been proved for both the PCD and LSC approaches in the single-phase case [165]. For PCD they are h-independent while for LSC the known bounds depend on h. Norm-equivalence of the PCD preconditioner is shown in [141]. While rigorous GMRES convergence bounds based on the field-of-values are also given in [141] for a mass matrix Schur complement approximation, to the best of our knowledge the final steps required to show such rigorous bounds for PCD are missing from the literature. Note that h-independent field-of-values bounds have been derived in [22] for the augmented Lagrangian preconditioner. While the analysis of preconditioned iterative methods for nonsymmetric problems is challenging, further work is required if h-independence of the approximate commutator preconditioners we have considered is to be proved. Moreover, analysis for our methods is further complicated by the variable coefficient nature of two-phase flow. Nonetheless, analysis for a Stokes interface problem is provided in [161] and may be built upon in order to further our understanding.

In terms of practical computation, we saw in Chapter 9 that additional features (that are not found in the basic two-phase flow model we have focused on) are often incorporated in order to model complex hydraulic processes more accurately. These features must be incorporated into the preconditioning methodology to provide more scalable solvers. It remains to be seen how this might be achieved. Further, testing on large-scale 3D problems of interest is required as well as in the setting of high performance computing on parallel architectures. Strong and weak scaling studies should be undertaken to assess the utility of our methods in the industrial setting along with a comparison of the wall clock times achieved with those of other methods.

A separate direction of future work is to further consider the multipreconditioning ideas discussed in Chapter 8. Our work suggests there may be potential for improved performance and that the approach might also be able to provide robustness in the solver methodology over the course of a simulations with disparate flow regimes. These ideas must be investigated further in order to evaluate the benefits which may be gained. Another course of enquiry is to examine adaptively selecting the choice of weighting parameter within the variant of sMPGMRES algorithm we considered. For example, it may be possible to use information on how successful the search directions obtained from the different preconditioners are in order to adaptively determine which preconditioner should be favoured through each stage of a simulation. This philosophy can apply in any case where it is unclear in advance which preconditioner is most favourable, not just in fluid flow simulations.

Finally, we emphasise that we did not explicitly require the two-phase nature in the construction of the preconditioners we have derived. Thus, the techniques we have introduced in this work might also be investigated for their utility when more general forms of variable density and variable viscosity flows are proposed.

Appendix A

Further numerical results

In this appendix we include additional tables of numerical results to help validate conclusions made in the thesis and provide further examples and evidence in order give a more complete picture than the selected results detailed in the main body of the thesis. These tables are referenced from within the thesis and, as such, the results are discussed there.

Table A.1: Average preconditioned GMRES iterations after Newton linearisation when using two-phase PCD (5.79) derived using a bi-weighted commutator, split two-phase PCD (5.64), and split two-phase PCD without incorporating appropriate boundary conditions (as in Section 5.2) for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the backward-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow) and varying grid size h, Reynolds number Re (of the water) and time-step Δt. Here the practical implementation uses the forms (5.89)–(5.90) as opposed to (5.92); cf. Table 5.2. Note that we omit the steady case since the results are identical to those in Table 5.2.

		Two-phase PCD (5.79)			Split two-phase PCD (5.64)			Split two-phase PCD without BCs		
Δt	$h \setminus Re$	10	100	500	10	100	500	10	100	500
$\frac{1}{10}$	$\frac{1}{4}$	15	16	22	15	16	24	22	24	33
	$\frac{1}{8}$	13	14	19	13	15	21	23	28	37
	$\frac{1}{16}$	14	13	18	14	14	20	28	36	47
	$\frac{1}{32}$	15	13	16	15	13	17	38	54	64
	$\frac{1}{64}$	15	14	15	16	14	15	54	79	98
1	$\frac{1}{4}$	21	19	22	21	21	27	26	28	37
	$\frac{1}{8}$	20	16	20	20	17	23	27	30	38
	$\frac{1}{16}$	21	17	18	21	17	19	31	38	45
	$\frac{1}{32}$	22	18	16	22	18	17	40	55	62
	$\frac{1}{64}$	22	18	17	22	18	17	57	81	96
10	$\frac{1}{4}$	23	24	33	23	25	36	28	34	48
	$\frac{1}{8}$	23	22	27	23	22	29	29	35	47
	$\frac{1}{16}$	24	23	23	24	23	24	33	44	52
	$\frac{1}{32}$	24	24	24	24	25	24	41	62	69
	$\frac{1}{64}$	24	25	24	24	26	25	58	91	104

Table A.2: Average preconditioned GMRES iterations after Newton linearisation when using the AMG iterated LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) using piecewise constant coefficients with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments. Iteration counts in bold show which method is most efficient.

Re	h	LSC(μ)	LSC($\mu^{1/2}$)	LSC($\mu, 1$)	LSC($\rho, 1$)	LSC(ρ, μ)	LSC($\rho, \mu^{1/2}$)
10	$\frac{1}{8}$	14	16	**13**	21	15	16
	$\frac{1}{16}$	15	16	**12**	19	16	16
	$\frac{1}{32}$	17	17	**13**	19	18	17
	$\frac{1}{64}$	25	21	**17**	21	24	20
$10^{1.5}$	$\frac{1}{8}$	16	18	**14**	23	17	18
	$\frac{1}{16}$	20	17	**15**	22	19	17
	$\frac{1}{32}$	21	22	**15**	23	21	20
	$\frac{1}{64}$	23	23	**19**	25	23	22
100	$\frac{1}{8}$	20	24	**19**	27	21	21
	$\frac{1}{16}$	23	22	**20**	27	21	21
	$\frac{1}{32}$	29	25	**22**	29	27	23
	$\frac{1}{64}$	34	29	28	31	29	**26**
$10^{2.5}$	$\frac{1}{8}$	**28**	37	**28**	34	**28**	**28**
	$\frac{1}{16}$	31	32	29	35	31	**27**
	$\frac{1}{32}$	36	35	32	37	33	**31**
	$\frac{1}{64}$	41	39	**32**	40	42	36
1000	$\frac{1}{8}$	52	72	60	59	**51**	52
	$\frac{1}{16}$	56	74	60	58	54	**51**
	$\frac{1}{32}$	**49**	72	53	58	**49**	51
	$\frac{1}{64}$	65	61	55	60	59	**51**

Table A.3: Average preconditioned GMRES iterations after Newton linearisation when using the AMG iterated LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) using a smoothed interface with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments. Iteration counts in bold show which method is most efficient.

Re	h	LSC(μ)	LSC($\mu^{1/2}$)	LSC($\mu, 1$)	LSC($\rho, 1$)	LSC(ρ, μ)	LSC($\rho, \mu^{1/2}$)
10	$\frac{1}{8}$	14	**9**	11	12	15	10
	$\frac{1}{16}$	15	12	**11**	15	19	14
	$\frac{1}{32}$	31	**15**	18	19	31	18
	$\frac{1}{64}$	40	**21**	**21**	25	41	27
$10^{1.5}$	$\frac{1}{8}$	15	**11**	12	15	16	12
	$\frac{1}{16}$	19	15	**13**	17	21	16
	$\frac{1}{32}$	31	**18**	19	21	32	20
	$\frac{1}{64}$	43	**21**	24	28	44	26
100	$\frac{1}{8}$	20	**17**	19	19	20	**17**
	$\frac{1}{16}$	27	**19**	22	21	27	20
	$\frac{1}{32}$	34	24	**23**	25	34	24
	$\frac{1}{64}$	44	29	**26**	32	48	30
$10^{2.5}$	$\frac{1}{8}$	**25**	27	26	27	26	**25**
	$\frac{1}{16}$	28	28	**27**	29	30	**27**
	$\frac{1}{32}$	41	31	33	32	41	**30**
	$\frac{1}{64}$	59	41	**35**	38	61	40
1000	$\frac{1}{8}$	**48**	59	57	56	**48**	50
	$\frac{1}{16}$	55	56	59	50	51	**44**
	$\frac{1}{32}$	64	54	55	49	63	**46**
	$\frac{1}{64}$	75	56	55	55	76	**54**

Table A.4: Average preconditioned GMRES iterations after Newton linearisation when using the AMG iterated LSC-type preconditioner (6.19) with different choices of scaling for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady lid driven cavity problem (Problem 1) using a fixed smoothed interface with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h. Note that iteration counts are given for the best choice of damping parameter ϵ used for boundary adjustments. Iteration counts in bold show which method is most efficient.

Re	h	LSC(μ)	LSC($\mu^{1/2}$)	LSC($\mu, 1$)	LSC($\rho, 1$)	LSC(ρ, μ)	LSC($\rho, \mu^{1/2}$)
10	$\frac{1}{8}$	13	12	**10**	51	17	18
	$\frac{1}{16}$	15	12	**11**	15	19	14
	$\frac{1}{32}$	16	16	**13**	20	21	17
	$\frac{1}{64}$	22	**15**	19	24	24	19
$10^{1.5}$	$\frac{1}{8}$	16	14	**13**	53	21	20
	$\frac{1}{16}$	19	15	**13**	17	21	16
	$\frac{1}{32}$	18	**16**	**16**	22	21	18
	$\frac{1}{64}$	**17**	**17**	18	26	21	21
100	$\frac{1}{8}$	19	18	**17**	56	23	24
	$\frac{1}{16}$	27	**19**	22	21	27	20
	$\frac{1}{32}$	24	**20**	**20**	26	27	22
	$\frac{1}{64}$	27	**21**	25	30	29	24
$10^{2.5}$	$\frac{1}{8}$	**25**	28	26	58	31	32
	$\frac{1}{16}$	28	28	**27**	29	30	**27**
	$\frac{1}{32}$	31	**29**	**29**	33	34	**29**
	$\frac{1}{64}$	37	**32**	33	36	38	34
1000	$\frac{1}{8}$	**48**	59	57	70	51	54
	$\frac{1}{16}$	55	56	59	50	51	**44**
	$\frac{1}{32}$	48	53	54	50	48	**45**
	$\frac{1}{64}$	49	53	55	51	49	**45**

Table A.5: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the LSC and PCD preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2\times10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), $h = 1/64$, and varying Reynolds number Re and time-step Δt.

		Weighted sMPGMRES Preconditioner										
Δt	Re	LSC	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	PCD
$\frac{1}{10}$	10	**10**	11	11	11	11	13	15	19	35	31	15
	$10^{1.5}$	**11**	12	**11**	**11**	**11**	12	16	16	22	32	15
	100	13	13	**12**	**12**	**12**	**12**	13	16	17	18	14
	$10^{2.5}$	15	14	13	**12**	**12**	**12**	**12**	13	14	16	14
	1000	17	14	**13**	**13**	**13**	**13**	14	14	14	16	16
1	10	14	14	14	**13**	**13**	14	16	24	48	39	17
	$10^{1.5}$	14	**13**	**13**	**13**	14	16	20	33	51	40	19
	100	**14**	**14**	**14**	15	17	20	25	44	56	40	23
	$10^{2.5}$	15	**14**	**14**	16	19	22	26	34	49	52	25
	1000	20	18	**17**	**17**	**17**	19	20	23	25	30	28
10	10	**14**	15	15	**14**	**14**	15	16	24	39	39	18
	$10^{1.5}$	**16**	**16**	**16**	**16**	**16**	18	21	37	49	40	22
	100	**19**	**19**	20	21	23	24	29	54	62	49	28
	$10^{2.5}$	26	**24**	**24**	26	28	38	38	62	79	76	34
	1000	40	**32**	**32**	38	36	41	45	62	73	99	43

Table A.6: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the PCD and SIMPLE-like preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), $h = 1/64$, and varying Reynolds number Re and time-step Δt.

		Weighted sMPGMRES Preconditioner										
Δt	Re	PCD	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	SIMPLE
$\frac{1}{10}$	10	15	**14**	**14**	**14**	15	17	21	31	57	96	124
	$10^{1.5}$	15	**14**	**14**	**14**	15	17	24	33	56	85	112
	100	**14**	**14**	**14**	15	15	17	23	40	55	68	93
	$10^{2.5}$	**14**	**14**	15	15	16	19	23	33	58	66	85
	1000	**16**	**16**	17	17	18	21	28	47	47	76	81
1	10	17	**16**	**16**	17	18	21	25	39	80	90	137
	$10^{1.5}$	19	**18**	**18**	**18**	19	23	28	40	91	101	144
	100	23	21	**20**	**20**	22	25	29	37	50	109	146
	$10^{2.5}$	25	24	**23**	24	25	28	34	43	51	68	144
	1000	28	**25**	26	26	28	31	39	50	56	79	139
10	10	18	**17**	**17**	18	18	22	25	37	72	113	138
	$10^{1.5}$	22	**20**	**20**	21	21	25	29	40	65	106	150
	100	28	26	**25**	**25**	26	29	33	41	56	120	165
	$10^{2.5}$	34	33	32	**30**	**30**	35	49	59	64	82	188
	1000	43	42	**40**	**40**	41	50	68	84	99	121	234

Table A.7: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the SIMPLE-like and PCD preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the lid driven cavity problem (Problem 1) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), $h = 1/64$, and varying Reynolds number Re and time-step Δt.

		Weighted sMPGMRES Preconditioner										
Δt	Re	SIMPLE	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	PCD
$\frac{1}{10}$	10	124	77	32	21	16	15	**14**	**14**	**14**	**14**	15
	$10^{1.5}$	112	61	30	18	15	15	**14**	**14**	**14**	**14**	15
	100	93	51	27	17	15	15	**14**	**14**	15	15	**14**
	$10^{2.5}$	85	44	24	17	16	15	15	15	15	15	**14**
	1000	81	36	22	19	18	18	18	19	19	19	**16**
1	10	137	68	38	23	19	**17**	**17**	**17**	**17**	18	**17**
	$10^{1.5}$	144	72	38	26	20	19	**18**	**18**	**18**	**18**	19
	100	146	77	37	25	22	21	**20**	**20**	21	21	23
	$10^{2.5}$	144	66	39	27	25	24	**23**	24	24	25	25
	1000	139	67	44	32	29	**27**	28	28	28	29	28
10	10	138	80	35	25	20	18	**17**	**17**	**17**	18	18
	$10^{1.5}$	150	71	40	27	22	21	**20**	**20**	22	**20**	22
	100	165	80	43	31	27	**25**	**25**	**25**	27	26	28
	$10^{2.5}$	188	90	53	39	32	**30**	31	31	32	32	34
	1000	234	125	89	58	43	**40**	**40**	41	43	42	43

Table A.8: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the PCD and LSC preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady backwards-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h.

		Weighted sMPGMRES Preconditioner										
Re	h	PCD	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	LSC
10	$\frac{1}{4}$	26	17	17	**16**	17	18	18	18	19	19	22
	$\frac{1}{8}$	24	16	15	14	14	**13**	**13**	**13**	**13**	14	14
	$\frac{1}{16}$	24	16	15	14	**13**	**13**	14	14	14	14	14
	$\frac{1}{32}$	23	16	16	15	15	16	**14**	**14**	**14**	**14**	**14**
100	$\frac{1}{4}$	31	22	21	**20**	**20**	**20**	22	25	28	31	27
	$\frac{1}{8}$	30	22	20	19	**18**	**18**	19	20	22	23	24
	$\frac{1}{16}$	30	22	20	**19**	**19**	**19**	**19**	20	21	21	23
	$\frac{1}{32}$	31	23	21	20	**19**	**19**	20	21	21	21	23
500	$\frac{1}{4}$	60	57	**53**	59	62	71	76	88	91	99	77
	$\frac{1}{8}$	55	**51**	53	57	63	78	77	99	117	129	73
	$\frac{1}{16}$	49	**41**	**41**	48	54	62	69	82	84	86	65
	$\frac{1}{32}$	47	36	**35**	36	43	44	52	57	61	61	63

Table A.9: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the LSC and SIMPLE-like preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady backwards-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h.

		Weighted sMPGMRES Preconditioner										
Re	h	LSC	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	SIMPLE
10	$\frac{1}{4}$	22	**16**	**16**	**16**	17	17	19	23	38	40	38
	$\frac{1}{8}$	14	**12**	**12**	13	13	14	16	17	24	31	58
	$\frac{1}{16}$	14	**13**	**13**	**13**	**13**	14	16	19	22	29	89
	$\frac{1}{32}$	14	**13**	**13**	**13**	14	14	16	20	23	33	137
100	$\frac{1}{4}$	27	24	24	23	**22**	25	29	47	68	69	55
	$\frac{1}{8}$	24	22	**20**	**20**	**20**	21	22	25	31	49	82
	$\frac{1}{16}$	23	21	**20**	**20**	21	21	22	24	26	33	121
	$\frac{1}{32}$	23	21	**20**	21	**20**	21	22	24	27	34	179
500	$\frac{1}{4}$	77	68	67	**66**	69	76	95	104	105	105	115
	$\frac{1}{8}$	73	**71**	75	76	78	93	121	168	274	275	190
	$\frac{1}{16}$	65	62	61	58	**57**	**57**	59	66	79	120	207
	$\frac{1}{32}$	63	58	55	52	50	**49**	50	50	56	66	284

Table A.10: Average preconditioned sMPGMRES iterations upon Newton linearisation using weighted combinations of the SIMPLE-like and LSC preconditioners for the $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ solution to the steady backwards-facing step problem (Problem 2) with density ratio $\widehat{\rho} = 1.2 \times 10^{-3}$, viscosity ratio $\widehat{\mu} = 1.8 \times 10^{-2}$ (values for air-water flow), and varying Reynolds number Re and grid size h.

		Weighted sMPGMRES Preconditioner										
Re	h	SIMPLE	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	LSC
10	$\frac{1}{4}$	38	36	22	18	**16**	**16**	**16**	**16**	**16**	**16**	22
	$\frac{1}{8}$	58	37	22	16	15	**13**	**13**	**13**	**13**	**13**	14
	$\frac{1}{16}$	89	35	21	17	14	14	14	**13**	**13**	**13**	14
	$\frac{1}{32}$	137	34	21	16	15	**14**	**14**	**14**	**14**	**14**	**14**
100	$\frac{1}{4}$	55	57	37	26	23	**22**	**22**	23	23	24	27
	$\frac{1}{8}$	82	51	30	24	22	21	**20**	21	21	22	24
	$\frac{1}{16}$	121	44	30	25	21	21	**20**	21	**20**	22	23
	$\frac{1}{32}$	179	40	29	23	21	20	20	**19**	20	21	23
500	$\frac{1}{4}$	115	103	93	75	66	**64**	68	67	70	70	77
	$\frac{1}{8}$	190	191	124	84	74	**68**	**68**	70	73	73	73
	$\frac{1}{16}$	207	129	68	60	57	**56**	58	58	58	61	65
	$\frac{1}{32}$	284	94	57	49	**48**	49	49	51	55	54	63

References

[1] M. Ainsworth and S. Sherwin. Domain decomposition preconditioners for p and hp finite element approximation of Stokes equations. *Comput. Methods Appl. Mech. Engrg.*, 175(3–4):243–266, 1999.

[2] I. Akkerman, Y. Bazilevs, C. E. Kees, and M. W. Farthing. Isogeometric analysis of free-surface flow. *J. Comput. Phys.*, 230(11):4137–4152, 2011.

[3] R. Amit, C. A. Hall, and T. A. Porsching. An application of network theory to the solution of implicit Navier–Stokes difference equations. *J. Comput. Phys.*, 40(1):183–201, 1981.

[4] D. M. Anderson, G. B. McFadden, and A. A. Wheeler. Diffuse-interface methods in fluid mechanics. *Annu. Rev. Fluid Mech.*, 30(1):139–165, 1998.

[5] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM J. Numer. Anal.*, 39(5):1749–1779, 2002.

[6] D. N. Arnold, F. Brezzi, and M. Fortin. A stable finite element for the Stokes equations. *Calcolo*, 21(4):337–344, 1984.

[7] O. Axelsson, X. He, and M. Neytcheva. Numerical solution of the time-dependent Navier–Stokes equation for variable density–variable viscosity. Part I. *Math. Model. Anal.*, 20(2):232–260, 2015.

[8] V. E. Badalassi, H. D. Ceniceros, and S. Banerjee. Computation of multiphase systems with phase field models. *J. Comput. Phys.*, 190(2):371–397, 2003.

[9] Z.-Z. Bai, G. H. Golub, L.-Z. Lu, and J.-F. Yin. Block triangular and skew-Hermitian splitting methods for positive-definite linear systems. *SIAM J. Sci. Comput.*, 26(3):844–863, 2005.

[10] Z.-Z. Bai, G. H. Golub, and M. K. Ng. Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems. *SIAM J. Matrix Anal. Appl.*, 24(3):603–626, 2003.

[11] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, D. A. May, L. Curfman McInnes, R. T. Mills, T. Munson, K. Rupp, P. Sanan, B. F. Smith, S. Zampini, H. Zhang, and H. Zhang. PETSc users Manual. Technical Report ANL-95/11 - Revision 3.10, Argonne National Laboratory, 2018. http://www.mcs.anl.gov/petsc.

[12] Y. Bazilevs and T. J. R. Hughes. Weak imposition of Dirichlet boundary conditions in fluid mechanics. *Comput. & Fluids*, 36(1):12–26, 2007.

[13] A. Bentley, N. Bootland, A. Wathen, and C. Kees. Implementation details of the level set two-phase Navier–Stokes equations in Proteus. Technical Report TR2017-10, Department of Mathematical Sciences, Clemson University, 2017. `http://www.clemson.edu/science/departments/mathematical-sciences/about/technical-reports.html`.

[14] M. Benzi. A generalization of the Hermitian and skew-Hermitian splitting iteration. *SIAM J. Matrix Anal. Appl.*, 31(2):360–374, 2009.

[15] M. Benzi, M. J. Gander, and G. H. Golub. Optimization of the Hermitian and skew-Hermitian splitting iteration for saddle-point problems. *BIT*, 43(5):881–900, 2003.

[16] M. Benzi and G. H. Golub. A preconditioner for generalized saddle point problems. *SIAM J. Matrix Anal. Appl.*, 26(1):20–41, 2004.

[17] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14(1):1–137, 2005.

[18] M. Benzi and X.-P. Guo. A dimensional split preconditioner for Stokes and linearized Navier–Stokes equations. *Appl. Numer. Math.*, 61(1):66–76, 2011.

[19] M. Benzi and J. Liu. An efficient solver for the incompressible Navier–Stokes equations in rotation form. *SIAM J. Sci. Comput.*, 29(5):1959–1981, 2007.

[20] M. Benzi, M. Ng, Q. Niu, and Z. Wang. A relaxed dimensional factorization preconditioner for the incompressible Navier–Stokes equations. *J. Comput. Phys.*, 230(16):6185–6202, 2011.

[21] M. Benzi and M. A. Olshanskii. An augmented Lagrangian-based approach to the Oseen problem. *SIAM J. Sci. Comput.*, 28(6):2095–2113, 2006.

[22] M. Benzi and M. A. Olshanskii. Field-of-values convergence analysis of augmented Lagrangian preconditioners for the linearized Navier–Stokes problem. *SIAM J. Numer. Anal.*, 49(2):770–788, 2011.

[23] M. Benzi, M. A. Olshanskii, and Z. Wang. Modified augmented Lagrangian preconditioners for the incompressible Navier–Stokes equations. *Int. J. Numer. Meth. Fluids*, 66(4):486–508, 2011.

[24] M. Benzi and A. J. Wathen. Some preconditioning techniques for saddle point problems. In W. H. A. Schilders, H. A. van der Vorst, and J. Rommes, editors, *Model Order Reduction: Theory, Research Aspects and Applications*, volume 13, pages 195–211. Springer, 2008.

[25] P. B. Bochev, C. R. Dohrmann, and M. D. Gunzburger. Stabilization of low-order mixed finite elements for the Stokes equations. *SIAM J. Numer. Anal.*, 44(1):82–101, 2006.

[26] A. Bonito, J.-L. Guermond, and S. Lee. Modified pressure-correction projection methods: Open boundary and variable time stepping. In A. Abdulle, S. Deparis, D. Kressner, F. Nobile, and M. Picasso, editors, *Numerical Mathematics and Advanced Applications - ENUMATH 2013. Lecture Notes in Computational Science and Engineering, Vol. 103*, pages 623–631. Springer, 2015.

[27] N. Bootland, A. Bentley, C. Kees, and A. Wathen. Preconditioners for two-phase incompressible Navier–Stokes flow. *Submitted to SIAM J. Sci. Comput.*, 2017.

[28] M. A. Botchev and G. H. Golub. A class of nonsymmetric preconditioners for saddle point problems. *SIAM J. Matrix Anal. Appl.*, 27(4):1125–1149, 2006.

[29] J. Boyle, M. Mihajlović, and J. Scott. HSL_MI20: An efficient AMG preconditioner for finite element problems in 3D. *Int. J. Numer. Meth. Engng.*, 82(1):64–98, 2010.

[30] J. U. Brackbill, D. B. Kothe, and C. Zemach. A continuum method for modeling surface tension. *J. Comput. Phys.*, 100(2):335–354, 1992.

[31] D. Braess and W. Dahmen. A cascadic multigrid algorithm for the Stokes equations. *Numer. Math.*, 82(2):179–191, 1999.

[32] D. Braess and R. Sarazin. An efficient smoother for the Stokes problem. *Appl. Numer. Math.*, 23(1):3–19, 1997.

[33] J. H. Bramble and J. E. Pasciak. A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Math. Comp.*, 50(181):1–17, 1988.

[34] J. H. Bramble and J. E. Pasciak. Iterative techniques for time dependent Stokes problems. *Comput. Math. Appl.*, 33(1–2):13–30, 1997.

[35] J. H. Bramble, J. E. Pasciak, and A. T. Vassilev. Uzawa type algorithms for nonsymmetric saddle point problems. *Math. Comp.*, 69(230):667–689, 2000.

[36] A. Brandt and N. Dinar. Multigrid solutions to elliptic flow problems. In S. Parter, editor, *Numerical Methods for Partial Differential Equations*, pages 53–147. Academic Press, 1979.

[37] F. Brezzi, J. Douglas, and L. D. Marini. Two families of mixed finite elements for second order elliptic problems. *Numer. Math.*, 47(2):217–235, 1985.

[38] R. Bridson and C. Greif. A multipreconditioned conjugate gradient algorithm. *SIAM J. Matrix Anal. Appl.*, 27(4):1056–1068, 2006.

[39] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM, second edition, 2000.

[40] M. O. Bristeau, R. Glowinski, and J. Periaux. Numerical methods for the Navier–Stokes equations. Applications to the simulation of compressible and incompressible viscous flows. *Comput. Phys. Rep.*, 6(1–6):73–187, 1987.

[41] C. Burstedde, O. Ghattas, G. Stadler, T. Tu, and L. C. Wilcox. Parallel scalable adjoint-based adaptive solution of variable-viscosity Stokes flow problems. *Comput. Methods Appl. Mech. Engrg.*, 198(21):1691–1700, 2009.

[42] J. Cahouet and J.-P. Chabard. Some fast 3D finite element solvers for the generalized Stokes problem. *Int. J. Numer. Meth. Fluids*, 8(8):869–895, 1988.

[43] M. Cai, A. Nonaka, J. B. Bell, B. E. Griffith, and A. Donev. Efficient variable-coefficient finite-volume Stokes solvers. *Commun. Comput. Phys.*, 16(5):1263–1297, 2014.

[44] Y. C. Chang, T. Y. Hou, B. Merriman, and S. Osher. A level set formulation of Eulerian interface capturing methods for incompressible fluid flows. *J. Comput. Phys.*, 124(2):449–464, 1996.

[45] J. Chessa and T. Belytschko. An extended finite element method for two-phase fluids. *J. Appl. Mech.*, 70(1):10–17, 2003.

[46] A. J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comp.*, 22(104):745–762, 1968.

[47] E. Chow and Y. Saad. Experimental study of ILU preconditioners for indefinite matrices. *J. Comput. Appl. Math.*, 86(2):387–414, 1997.

[48] P. Ciarlet, Jr., J. Huang, and J. Zou. Some observations on generalized saddle-point problems. *SIAM J. Matrix Anal. Appl.*, 25(1):224–236, 2003.

[49] B. Cockburn, G. Kanschat, and D. Schötzau. A locally conservative LDG method for the incompressible Navier-Stokes equations. *Math. Comp.*, 74(251):1067–1095, 2005.

[50] R. Codina. A stabilized finite element method for generalized stationary incompressible flows. *Comput. Methods Appl. Mech. Engrg.*, 190(20–21):2681–2706, 2001.

[51] A. Collagrossi and M. Landrini. Numerical simulation of interfacial flows by smoothed particle hydrodynamics. *J. Comput. Phys.*, 191(2):448–475, 2003.

[52] M. Crouzeix. On an operator related to the convergence of Uzawa's algorithm for the Stokes equation. In M.-O. Bristeau, G. Etgen, W. Fitzgibbon, J. Lions, J. Périaux, and M. Wheeler, editors, *Computational Science for the 21st-Century*, pages 242–249. Wiley, Chichester, 1997.

[53] M. Crouzeix and P.-A. Raviart. Conforming and nonconforming finite element methods for solving the stationary Stokes equations I. *Rev. Française Automat. Informat. Recherche Opérationnelle. Mathématique*, 7(R3):33–75, 1973.

[54] E. C. Cyr, J. N. Shadid, and R. S. Tuminaro. Stabilization and scalable block preconditioning for the Navier–Stokes equations. *J. Comput. Phys.*, 231(2):345–363, 2012.

[55] L. D. Dalcin, R. R. Paz, P. A. Kler, and A. Cosimo. Parallel distributed computing using Python. *Adv. Water Resour.*, 34(9):1124–1139, 2011.

[56] E. de Sturler and J. Liesen. Block-diagonal and constraint preconditioners for nonsymmetric indefinite linear systems. Part I: Theory. *SIAM J. Sci. Comput.*, 26(5):1598–1619, 2005.

[57] M. S. Dodd and A. Ferrante. A fast pressure-correction method for incompressible two-fluid flows. *J. Comput. Phys.*, 273:416–434, 2014.

[58] C. R. Dohrmann and P. B. Bochev. A stabilized finite element method for the Stokes problem based on polynomial pressure projections. *Int. J. Numer. Meth. Fluids*, 46(2):183–201, 2004.

[59] V. Dolean, F. Nataf, and G. Rapin. Deriving a new domain decomposition method for the Stokes equations using the Smith factorization. *Math. Comp.*, 78(266):789–814, 2009.

[60] H. S. Dollar and A. J. Wathen. Approximate factorization constraint preconditioners for saddle-point matrices. *SIAM J. Sci. Comput.*, 27(5):1555–1572, 2006.

[61] S. Dong and J. Shen. A time-stepping scheme involving constant coefficient matrices for phase-field simulations of two-phase incompressible flows with large density ratios. *J. Comput. Phys.*, 231(17):5788–5804, 2012.

[62] S. Dong and X. Wang. A rotational pressure-correction scheme for incompressible two-phase flows with open boundaries. *PLoS ONE*, 11(5):e0154565, 2016.

[63] H. Elman, V. E. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro. Block preconditioners based on approximate commutators. *SIAM J. Sci. Comput.*, 27(5):1651–1668, 2006.

[64] H. Elman, V. E. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro. A taxonomy and comparison of parallel block multi-level preconditioners for the incompressible Navier–Stokes equations. *J. Comput. Phys.*, 227(3):1790–1808, 2008.

[65] H. Elman, V. E. Howle, J. Shadid, D. Silvester, and R. Tuminaro. Least squares preconditioners for stabilized discretizations of the Navier–Stokes equations. *SIAM J. Sci. Comput.*, 30(1):290–311, 2007.

[66] H. C. Elman. Multigrid and Krylov subspace methods for the discrete Stokes equations. *Int. J. Numer. Meth. Fluids*, 22(8):755–770, 1996.

[67] H. C. Elman. Preconditioning for the steady-state Navier–Stokes equations with low viscosity. *SIAM J. Sci. Comput.*, 20(4):1299–1316, 1999.

[68] H. C. Elman and G. H. Golub. Inexact and preconditioned Uzawa algorithms for saddle point problems. *SIAM J. Numer. Anal.*, 31(6):1645–1661, 1994.

[69] H. C. Elman, D. Loghin, and A. J. Wathen. Preconditioning techniques for Newton's method for the incompressible Navier–Stokes equations. *BIT*, 43(5):961–974, 2003.

[70] H. C. Elman, A. Ramage, and D. J. Silvester. Algorithm 866: IFISS, a Matlab toolbox for modelling incompressible flow. *ACM Trans. Math. Software*, 33(2), 2007.

[71] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics*. Oxford University Press, second edition, 2014.

[72] H. C. Elman and R. S. Tuminaro. Boundary conditions in approximate commutator preconditioners for the Navier–Stokes equations. *Electron. Trans. Numer. Anal.*, 35:257–280, 2009.

[73] M. Embree. The tortoise and the hare restart GMRES. *SIAM Rev.*, 45(2):259–266, 2003.

[74] H.-T. Fan and X.-Y. Zhu. A modified relaxed splitting preconditioner for generalized saddle point problems from the incompressible Navier–Stokes equations. *Appl. Math. Lett.*, 55:18–26, 2016.

[75] H.-T. Fan, X.-Y. Zhu, and B. Zheng. The generalized double shift-splitting preconditioner for nonsymmetric generalized saddle point problems from the steady Navier–Stokes equations. *Comp. Appl. Math.*, 37(3):3256–3266, 2018.

[76] M. Fortin and R. Glowinski. *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*. North-Holland, 1983.

[77] R. W. Freund and N. M. Nachtigal. QMR: A quasi-minimal residual method for non-Hermitian linear systems. *Numer. Math.*, 60(1):315–339, 1991.

[78] A. Gauthier, F. Saleri, and A. Veneziani. A fast preconditioner for the incompressible Navier Stokes equations. *Comput. Visual Sci.*, 6(2–3):105–112, 2004.

[79] T. Gelhard, G. Lube, M. A. Olshanskii, and J.-H. Starcke. Stabilized finite element schemes with LBB-stable elements for incompressible flows. *J. Comput. Appl. Math.*, 177(2):243–267, 2005.

[80] V. Girault and P.-A. Raviart. *Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms*. Springer, 1986.

[81] G. H. Golub and R. S. Varga. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods. *Numer. Math.*, 3(1):147–156, 1961.

[82] G. H. Golub and A. J. Wathen. An iteration for indefinite systems and its application to the Navier–Stokes equations. *SIAM J. Sci. Comput.*, 19(2):530–539, 1998.

[83] N. Gould, D. Orban, and T. Rees. Projected Krylov methods for saddle-point systems. *SIAM J. Matrix Anal. Appl.*, 35(4):1329–1343, 2014.

[84] I. G. Graham, P. O. Lechner, and R. Scheichl. Domain decomposition for multiscale PDEs. *Numer. Math.*, 106(4):589–626, 2007.

[85] I. G. Graham and R. Scheichl. Robust domain decomposition algorithms for multiscale PDEs. *Numer. Methods Partial Differential Equations*, 23(4):859–878, 2007.

[86] I. G. Graham, A. Spence, and E. Vainikko. Parallel iterative methods for Navier–Stokes equations and application to eigenvalue computation. *Concurrency Computat. Pract. Exper.*, 15(11-12):1151–1168, 2003.

[87] A. Greenbaum, V. Pták, and Z. Strakoš. Any nonincreasing convergence curve is possible for GMRES. *SIAM J. Matrix Anal. Appl.*, 17(3):465–469, 1996.

[88] C. Greif, T. Rees, and D. B. Szyld. GMRES with multiple preconditioners. *SeMA*, 74(2):213–231, 2017.

[89] B. E. Griffith. An accurate and efficient method for the incompressible Navier–Stokes equations using the projection method as a preconditioner. *J. Comput. Phys.*, 228(20):7565–7595, 2009.

[90] P. P. Grinevich and M. A. Olshanskii. An iterative method for the Stokes-type problem with variable viscosity. *SIAM J. Sci. Comput.*, 31(5):3959–3978, 2009.

[91] S. Groß, V. Reichelt, and A. Reusken. A finite element based level set method for two-phase incompressible flows. *Comput. Visual Sci.*, 9(4):239–257, 2006.

[92] S. Groß and A. Reusken. An extended pressure finite element space for two-phase incompressible flows with surface tension. *J. Comput. Phys.*, 224(1):40–58, 2007.

[93] S. Gross and A. Reusken. Finite element discretization error analysis of a surface tension force in two-phase incompressible flows. *SIAM. J. Numer. Anal.*, 45(4):1679–1700, 2007.

[94] J. L. Guermond, P. Minev, and J. Shen. An overview of projection methods for incompressible flows. *Comput. Methods Appl. Mech. Engrg.*, 195(44–47):6011–6045, 2006.

[95] J.-L. Guermond and A. Salgado. A splitting method for incompressible flows with variable density based on a pressure Poisson equation. *J. Comput. Phys.*, 228(8):2834–2846, 2009.

[96] A. Günnel, R. Herzog, and E. Sachs. A note on preconditioners and scalar products in Krylov subspace methods for self-adjoint problems in Hilbert space. *Electron. Trans. Numer. Anal.*, 41:13–20, 2014.

[97] C. A. Hall. Numerical solution of Navier–Stokes problems by the dual variable method. *SIAM J. Alg. Disc. Meth.*, 6(2):220–236, 1985.

[98] C. A. Hall and X. Ye. Construction of null bases for the divergence operator associated with incompressible Navier-Stokes equations. *Linear Algebra Appl.*, 171:9–52, 1992.

[99] F. H. Harlow and J. E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids*, 8(12):2182–2189, 1965.

[100] X. He and M. Neytcheva. Preconditioning the incompressible Navier–Stokes equations with variable viscosity. *J. Comput. Math.*, 30(5):461–482, 2012.

[101] X. He, M. Neytcheva, and C. Vuik. On preconditioning incompressible non-Newtonian flow problems. Technical Report 13-07, Delft University of Technology, Reports of the Delft Institute of Applied Mathematics, 2013.

[102] Y. He and S. P. MacLachlan. Local Fourier analysis of block-structured multigrid relaxation schemes for the Stokes equations. *Numer. Linear Algebra Appl.*, 25(3):e2147, 2018.

[103] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand.*, 49(1):409–436, 1952.

[104] R. Hiptmair. Operator preconditioning. *Comput. Math. Appl.*, 52(5):699–706, 2006.

[105] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *J. Comput. Phys.*, 39(1):201–225, 1981.

[106] R. M. Holland, A. J. Wathen, and G. J. Shaw. Sparse approximate inverses and target matrices. *SIAM J. Sci. Comput.*, 26(3):1000–1011, 2005.

[107] P. Hood and C. Taylor. Navier–Stokes equations using mixed interpolation. In J. T. Oden, R. H. Gallagher, O. C. Zienkiewicz, and C. Taylor, editors, *Finite Element Methods in Flow Problems*, pages 121–132. University of Alabama in Huntsville Press, 1974.

[108] B. Hosseini, S. Turek, M. Möller, and C. Palmes. Isogeometric analysis of the Navier–Stokes–Cahn–Hilliard equations with application to incompressible two-phase flows. *J. Comput. Phys.*, 348:171–194, 2017.

[109] T. J. R. Hughes. Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Comput. Methods Appl. Mech. Engrg.*, 127(1–4):387–401, 1995.

[110] F.-N. Hwang and X.-C. Cai. Parallel fully coupled Schwarz preconditioners for saddle point problems. *Electron. Trans. Numer. Anal.*, 22:146–162, 2006.

[111] I. C. F. Ipsen. A note on preconditioning nonsymmetric matrices. *SIAM J. Sci. Comput.*, 23(3):1050–1051, 2001.

[112] D. Jacqmin. Calculation of two-phase Navier–Stokes flows using phase-field modeling. *J. Comput. Phys.*, 155(1):96–127, 1999.

[113] A. Janka. Smoothed aggregation multigrid for a Stokes problem. *Comput. Visual Sci.*, 11(3):169–180, 2008.

[114] M. Jobelin, C. Lapuerta, J.-C. Latché, Ph. Angot, and B. Piar. A finite element penalty–projection method for incompressible flows. *J. Comput. Phys.*, 217(2):502–518, 2006.

[115] V. John, A. Linke, C. Merdon, M. Neilan, and L. G. Rebholz. On the divergence constraint in mixed finite element methods for incompressible flows. *SIAM Rev.*, 59(3):492–544, 2017.

[116] V. John and L. Tobiska. Numerical performance of smoothers in coupled multigrid methods for the parallel solution of the incompressible Navier–Stokes equations. *Int. J. Numer. Meth. Fluids*, 33(4):453–473, 2000.

[117] O. A. Karakashian. On a Galerkin–Lagrange multiplier method for the stationary Navier–Stokes equations. *SIAM J. Numer. Anal.*, 19(5):909–923, 1982.

[118] D. Kay, D. Loghin, and A. Wathen. A preconditioner for the steady-state Navier–Stokes equations. *SIAM J. Sci. Comput.*, 24(1):237–256, 2002.

[119] Y.-F. Ke and C.-F. Ma. The parameterized preconditioner for the generalized saddle point problems from the incompressible Navier–Stokes equations. *Comp. Appl. Math.*, 37(3):3385–3398, 2018.

[120] C. E. Kees, I. Akkerman, M. W. Farthing, and Y. Bazilevs. A conservative level set method suitable for variable-order approximations and unstructured meshes. *J. Comput. Phys.*, 230(12):4536–4558, 2011.

[121] C. Keller, N. I. M. Gould, and A. J. Wathen. Constraint preconditioning for indefinite linear systems. *SIAM J. Matrix Anal. Appl.*, 21(4):1300–1317, 2000.

[122] C. T. Kelley and D. E. Keyes. Convergence analysis of pseudo-transient continuation. *SIAM J. Numer. Anal.*, 35(2):508–523, 1998.

[123] K. Kergrene, I. Babuška, and U. Banerjee. Stable generalized finite element method and associated iterative schemes; application to interface problems. *Comput. Methods Appl. Mech. Engrg.*, 305(1–36):6011–6045, 2016.

[124] A. Klawonn and L. F. Pavarino. Overlapping Schwarz methods for mixed linear elasticity and Stokes problems. *Comput. Methods Appl. Mech. Engrg.*, 165(1–4):233–245, 1998.

[125] A. Klawonn and G. Starke. Block triangular preconditioners for nonsymmetric saddle point problems: field-of-values analysis. *Numer. Math.*, 81(4):577–594, 1999.

[126] I. N. Konshin, M. A. Olshanskii, and Y. V. Vassilevski. ILU preconditioners for nonsymmetric saddle-point matrices with application to the incompressible Navier–Stokes equations. *SIAM J. Sci. Comput.*, 37(5):A2171–A2197, 2015.

[127] L. A. Krukier, B. L. Krukier, and Z.-R. Ren. Generalized skew-Hermitian triangular splitting iteration methods for saddle-point linear systems. *Numer. Linear Algebra Appl.*, 21(1):152–170, 2014.

[128] P. Kyzyżanowski. On block preconditioners for nonsymmetric saddle point problems. *SIAM J. Sci. Comput.*, 23(1):157–169, 2001.

[129] O. A. Ladyzhenskaya. *The Mathematical Theory of Viscous Incompressible Flow*. Gordon and Breach Science Publishers, second English edition, 1969.

[130] J. Laminie and P. Poullet. A dynamic penalty or projection method for incompressible fluids. *J. Comput. Phys.*, 50(1):213–234, 2012.

[131] J. Li. A dual-primal FETI method for incompressible Stokes equations. *Numer. Math.*, 102(2):257–275, 2005.

[132] J. Li and X. Tu. A nonoverlapping domain decomposition method for incompressible Stokes equations with continuous pressures. *SIAM J. Numer. Anal.*, 51(2):1235–1253, 2013.

[133] J. Li and O. Widlund. BDDC algorithms for incompressible Stokes equations. *SIAM J. Numer. Anal.*, 44(6):2432–2455, 2006.

[134] X. S. Li and J. W. Demmel. SuperLU_DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Trans. Math. Software*, 29(2):110–140, 2003.

[135] X. S. Li, J. W. Demmel, J. R. Gilbert, L. Grigori, P. Sao, M. Shao, and I. Yamazaki. SuperLU Users' Guide. Technical Report LBNL-44289, Lawrence Berkeley National Laboratory, September 1999. `http://crd.lbl.gov/~xiaoye/SuperLU/`. Last update: March 2016.

[136] A. Limache, S. Idelsohn, R. Rossi, and E. Oñate. The violation of objectivity in Laplace formulations of the Navier–Stokes equations. *Int. J. Numer. Meth. Fluids*, 54(6–8):639–664, 2007.

[137] P. T. Lin, M. Sala, J. N. Shadid, and R. S. Tuminaro. Performance of fully coupled algebraic multilevel domain decomposition preconditioners for incompressible flow and transport. *Int. J. Numer. Meth. Engng.*, 67(2):208–225, 2006.

[138] D. Loghin. Analysis of preconditioned Picard iterations for the Navier–Stokes equations. Technical Report 01/10, Oxford University Numerical Analysis Group, 2001.

[139] D. Loghin and A. J. Wathen. Schur complement preconditioners for the Navier–Stokes equations. *Int. J. Numer. Meth. Fluids*, 40(3–4):403–412, 2002.

[140] D. Loghin and A. J. Wathen. Schur complement preconditioning for elliptic systems of partial differential equations. *Numer. Linear Algebra Appl.*, 10(5–6):423–443, 2003.

[141] D. Loghin and A. J. Wathen. Analysis of preconditioners for saddle-point problems. *SIAM J. Sci. Comput.*, 25(6):2029–2049, 2004.

[142] G. Lube, L. Müller, and F. C. Otto. A nonoverlapping domain decomposition method for stabilized finite element approximations of the Oseen equations. *J. Comput. Appl. Math.*, 132(2):211–236, 2001.

[143] G. Lube and M. A. Olshanskii. Stable finite-element calculation of incompressible flows using the rotation form of convection. *IMA J. Numer. Anal.*, 22(3):437–461, 2002.

[144] L. Luo, Q. Zhang, X.-P. Wang, and X.-C. Cai. A parallel two-phase flow solver on unstructured mesh in 3D. In C.-O. Lee, X.-C. Cai, D. E. Keyes, H. H. Kim, A. Klawonn, E.-J. Park, and O. B Widlund, editors, *Domain Decomposition Methods in Science and Engineering XXIII. Lecture Notes in Computational Science and Engineering, Vol. 116*, pages 379–387. Springer, 2017.

[145] Y. Maday, D. Meiron, A. T. Patera, and E. M. Rønquist. Analysis of iterative methods for the steady and unsteady Stokes problem: Application to spectral element discretizations. *SIAM J. Sci. Comput.*, 14(2):310–337, 1993.

[146] S. Manservisi. Numerical analysis of Vanka-type solvers for steady Stokes and Navier–Stokes flows. *SIAM J. Numer. Anal.*, 44(5):2025–2056, 2006.

[147] T. A. Manteuffel and S. V. Parter. Preconditioning and boundary conditions. *SIAM J. Numer. Anal.*, 27(3):656–694, 1990.

[148] K.-A. Mardal and R. Winther. Preconditioning discretizations of systems of partial differential equations. *Numer. Linear Algebra Appl.*, 18(1):1–40, 2011.

[149] D. A. May, J. Brown, and L. Le Pourhiet. A scalable, matrix-free multigrid preconditioner for finite element discretizations of heterogeneous Stokes flow. *Comput. Methods Appl. Mech. Engrg.*, 290:492–523, 2015.

[150] D. A. May and L. Moresi. Preconditioned iterative methods for Stokes flow problems arising in computational geodynamics. *Phys. Earth Planet. Inter.*, 171(1):33–47, 2008.

[151] S. A. Melchior, V. Legat, P. Van Dooren, and A. J. Wathen. Analysis of preconditioned iterative solvers for incompressible flow problems. *Int. J. Numer. Meth. Fluids*, 68(3):269–286, 2012.

[152] M. Mihajlović. private communication, 2016.

[153] M. F. Murphy, G. H. Golub, and A. J. Wathen. A note on preconditioning for indefinite linear systems. *SIAM J. Sci. Comput.*, 21(6):1969–1972, 2000.

[154] J.-C. Nédélec. Mixed finite elements in $\mathbb{R}^3$. *Numer. Math.*, 35(3):315–341, 1980.

[155] J.-C. Nédélec. A new family of mixed finite elements in $\mathbb{R}^3$. *Numer. Math.*, 50(1):57–81, 1986.

[156] M.-J. Ni. Consistent projection methods for variable density incompressible Navier–Stokes equations with continuous surface forces on a rectangular collocated mesh. *J. Comput. Phys.*, 228(18):6938–6956, 2009.

[157] D. P. O'Leary and R. E. White. Multi-splittings of matrices and parallel solution of linear systems. *SIAM J. Algebraic Discret. Methods*, 6(4):630–640, 1985.

[158] M. A. Olshanskii. An iterative solver for the Oseen problem and numerical solution of incompressible Navier–Stokes equations. *Numer. Linear Algebra Appl.*, 6(5):353–378, 1999.

[159] M. A. Olshanskii. A low order Galerkin finite element method for the Navier–Stokes equations of steady incompressible flow: a stabilization issue and iterative methods. *Comput. Methods Appl. Mech. Engrg.*, 191(47–48):5515–5536, 2002.

[160] M. A. Olshanskii. Multigrid analysis for the time dependent Stokes problem. *Math. Comp.*, 81(277):57–79, 2012.

[161] M. A. Olshanskii, J. Peters, and A. Reusken. Uniform preconditioners for a parameter dependent saddle point problem with application to generalized Stokes interface equations. *Numer. Math.*, 105(1):159–191, 2006.

[162] M. A. Olshanskii and A. Reusken. Navier–Stokes equations in rotation form: a robust multigrid solver for the velocity problem. *SIAM J. Sci. Comput.*, 23(5):1683–1706, 2002.

[163] M. A. Olshanskii and A. Reusken. Convergence analysis of a multigrid method for a convection-dominated model problem. *SIAM J. Numer. Anal.*, 42(3):1261–1291, 2004.

[164] M. A. Olshanskii and A. Reusken. Analysis of a Stokes interface problem. *Numer. Math.*, 103(1):129–149, 2006.

[165] M. A. Olshanskii and Y. V. Vassilevski. Pressure Schur complement preconditioners for the discrete Oseen problem. *SIAM J. Sci. Comput.*, 29(6):2686–2794, 2007.

[166] C. W. Oosterlee and F. J. Gaspar Lorenz. Multigrid methods for the Stokes system. *Comput. Sci. Eng.*, 8(6):34–43, 2006.

[167] P. Oswald. An optimal multilevel preconditioner for solenoidal approximations of the two-dimensional Stokes problem. *IMA J. Numer. Anal.*, 18(2):207–228, 1998.

[168] F.-C. Otto and G. Lube. A nonoverlapping domain decomposition method for the Oseen equations. *Math. Models Methods Appl.*, 8(6):1091–1117, 1998.

[169] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975.

[170] S. Patankar. *Numerical Heat Transfer and Fluid Flow.* McGraw-Hill, 1980.

[171] S. V. Patankar and D. B. Spalding. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *Int. J. Heat Mass Transfer*, 15(10):1787–1806, 1972.

[172] L. F. Pavarino. Indefinite overlapping Schwarz methods for time-dependent Stokes problems. *Comput. Methods Appl. Mech. Engrg.*, 187(1–2):35–51, 2000.

[173] L. F. Pavarino and O. B. Widlund. Balancing Neumann-Neumann methods for incompressible Stokes equations. *Comm. Pure Appl. Math.*, 55(3):302–335, 2002.

[174] M. Pernice. A hybrid multigrid method for the steady-state incompressible Navier-Stokes equations. *Electron. Trans. Numer. Anal.*, 10:74–91, 2000.

[175] J. B. Perot. An analysis of the fractional step method. *J. Comput. Phys.*, 108(1):51–58, 1993.

[176] J. Pestana and T. Rees. Null-space preconditioners for saddle point systems. *SIAM J. Matrix Anal. Appl.*, 37(3):1103–1128, 2016.

[177] J. Pestana and A. J. Wathen. Combination preconditioning of saddle point systems for positive definiteness. *Numer. Linear Algebra Appl.*, 20(5):785–808, 2013.

[178] J. Peters, V. Reichelt, and A. Reusken. Fast iterative solvers for discrete Stokes equations. *SIAM J. Sci. Comput.*, 27(2):646–666, 2005.

[179] J. E. Pilliod Jr. and E. G. Puckett. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *J. Comput. Phys.*, 199(2):465–502, 2004.

[180] A. Prohl. On pressure approximation via projection methods for nonstationary incompressible Navier–Stokes equations. *SIAM J. Numer. Anal.*, 47(1):158–180, 2008.

[181] J.-H. Pyo and J. Shen. Gauge–Uzawa methods for incompressible flows with variable density. *J. Comput. Phys.*, 221(1):181–197, 2007.

[182] A. Quarteroni, F. Saleri, and A. Veneziani. Factorization methods for the numerical approximation of Navier–Stokes equations. *Comput. Methods Appl. Mech. Engrg.*, 188(1–3):505–526, 2000.

[183] A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations*. Oxford University Press, 1999.

[184] P.-A. Raviart and J. M. Thomas. A mixed finite element method for 2-nd order elliptic problems. In I. Galligani and E. Magenes, editors, *Mathematical Aspects of Finite Element Methods. Lecture Notes in Mathematics, Vol. 606*, pages 292–315. Springer, 1977.

[185] M. Raw. A coupled algebraic multigrid method for the 3D Navier-Stokes equations. In W. Hackbusch and G. Wittum, editors, *Fast Solvers for Flow Problems,* Notes on Numerical Fluid Mechanics (NNFM), pages 204–215. Vieweg+Teubner Verlag, 1995.

[186] L. G. Rebholz and M. Xiao. Improved accuracy in algebraic splitting methods for Navier–Stokes equations. *SIAM J. Sci. Comput.*, 39(4):A1489–A1513, 2017.

[187] T. Rees and J. Scott. A comparative study of null-space factorizations for sparse symmetric saddle point systems. *Numer. Linear Algebra Appl.*, 25(1):e2103, 2018.

[188] D. Rempfer. On boundary conditions for incompressible Navier-Stokes problems. *Appl. Mech. Rev.*, 59(3):107–125, 2006.

[189] H.-G. Roos, M. Stynes, and L. Tobiska. *Numerical Methods for Singularly Perturbed Differential Equations: Convection-Diffusion and Flow Problems*. Springer, 1996.

[190] J. Rudi, G. Stadler, and O. Ghattas. Weighted BFBT preconditioner for Stokes flow problems with highly heterogeneous viscosity. *SIAM J. Sci. Comput.*, 39(5):S272–S297, 2017.

[191] M. Rudman. Volume-tracking methods for interfacial flow calculations. *Int. J. Numer. Meth. Fluids*, 24(7):671–691, 1997.

[192] P. L. Rui, H. Yong, and R. S. Chen. Multipreconditioned GMRES method for electromagnetic wave scattering problems. *Microw. Opt. Technol. Lett.*, 50(1):150–152, 2007.

[193] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM J. Sci. Comput.*, 14(2):461–469, 1993.

[194] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, second edition, 2003.

[195] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, 1986.

[196] F. Saleri and A. Veneziani. Pressure correction algebraic splitting methods for the incompressible Navier–Stokes equations. *SIAM J. Numer. Anal.*, 43(1):174–194, 2005.

[197] V. Sarin and A. Sameh. An efficient iterative method for the generalized Stokes problem. *SIAM J. Sci. Comput.*, 19(1):206–226, 1998.

[198] R. Scardovelli and S. Zaleski. Direct numerical simulation of free-surface and interfacial flow. *Annu. Rev. Fluid Mech.*, 31(1):567–603, 1999.

[199] R. Scheichl and E. Vainikko. Additive Schwarz with aggregation-based coarsening for elliptic problems with highly variable coefficients. *Computing*, 80(4):319–343, 2007.

[200] R. Scheichl, P. S. Vassilevski, and L. T. Zikatanov. Multilevel methods for elliptic problems with highly varying coefficients on nonaligned coarse grids. *SIAM J. Numer. Anal.*, 50(3):1675–1694, 2012.

[201] J. Schöberl. Multigrid methods for a parameter dependent problem in primal variables. *Numer. Math.*, 84(1):97–119, 1999.

[202] J. A. Sethian and P. Smereka. Level set methods for fluid interfaces. *Annu. Rev. Fluid Mech.*, 35(1):341–372, 2003.

[203] J. N. Shadid, R. S. Tuminaro, K. D. Devine, G. L. Hennigan, and P. T. Lin. Performance of fully coupled domain decomposition preconditioners for finite element transport/reaction simulations. *J. Comput. Phys.*, 205(1):24–47, 2005.

[204] G. J. Shaw and S. Sivaloganathan. On the smoothing properties of the SIMPLE pressure-correction algorithm. *Int. J. Numer. Meth. Fluids*, 8(4):441–461, 1988.

[205] H. Shen and H. Xiang. Uzawa algorithms with variable relaxation for non-symmetric generalized saddle point problems. *Numer. Linear Algebra Appl.*, 22(6):1020–1038, 2015.

[206] J. Shen and X. Yang. A phase-field model and its numerical approximation for two-phase incompressible flows with different densities and viscosities. *SIAM J. Sci. Comput.*, 32(3):1159–1179, 2010.

[207] D. Silvester, H. Elman, D. Kay, and A. Wathen. Efficient preconditioning of the linearized Navier–Stokes equations for incompressible flow. *J. Comput. Appl. Math.*, 128(1):261–279, 2001.

[208] D. Silvester and A. Wathen. Fast iterative solution of stabilised Stokes systems part II: using general block preconditioners. *SIAM J. Numer. Anal.*, 31(5):1352–1367, 1994.

[209] D. J. Silvester, H. C. Elman, and A. Ramage. Incompressible Flow and Iterative Solver Software (IFISS), version 3.5, September 2016. http://www.manchester.ac.uk/ifiss/.

[210] V. Simoncini and M. Benzi. Spectral properties of the Hermitian and skew-Hermitian splitting preconditioner for saddle point problems. *SIAM J. Matrix Anal. Appl.*, 26(2):377–389, 2004.

[211] V. Simoncini and D. B. Szyld. Recent computational developments in Krylov subspace methods for linear systems. *Numer. Linear Algebra Appl.*, 14(1):1–59, 2007.

[212] P. Sonneveld and M. B. van Gijzen. IDR(s): A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations. *SIAM J. Sci. Comput.*, 31(2):1035–1062, 2008.

[213] G. Starke. Field-of-values analysis of preconditioned iterative methods for non-symmetric elliptic problems. *Numer. Math.*, 78(1):103–117, 1997.

[214] M. Stoll and A. Wathen. Combination preconditioning and the Bramble–Pasciak$^+$ preconditioner. *SIAM J. Matrix Anal. Appl.*, 30(2):582–608, 2008.

[215] M. Stoll and A. Wathen. All-at-once solution of time-dependent Stokes control. *J. Comput. Phys.*, 232(1):498–515, 2013.

[216] L.-Y. Sun and J. Liu. Constraint preconditioning for nonsymmetric indefinite linear systems. *Numer. Linear Algebra Appl.*, 17(4):677–689, 2010.

[217] P. Sun, G. Xue, C. Wang, and J. Xu. A domain decomposition method for two-phase transport model in the cathode of a polymer electrolyte fuel cell. *J. Comput. Phys.*, 228(16):6016–6036, 2009.

[218] M. Sussman and E. G. Puckett. A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows. *J. Comput. Phys.*, 162(2):301–337, 2000.

[219] Syamsudhuha and D. J. Silvester. Efficient solution of the steady-state Navier–Stokes equations using a multigrid preconditioned Newton–Krylov method. *Int. J. Numer. Meth. Fluids*, 43(12):1407–1427, 2003.

[220] R. Temam. Sur l'approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires (II). *Art. Rat. Mech. Anal.*, 33(5):377–385, 1969.

[221] R. Temam. *Navier–Stokes Equations*. North-Holland, 1979.

[222] R. Temam. *Navier–Stokes Equations and Nonlinear Functional Analysis*. SIAM, 1983.

[223] T. E. Tezduyar. Stabilized finite element formulations for incompressible flow computations. *Adv. Appl. Mech.*, 28:1–44, 1992.

[224] A.-K. Tornberg and B. Engquist. A finite element based level-set method for multiphase flow applications. *Comput. Visual Sci.*, 3(1–2):93–101, 2000.

[225] A. Toselli and O. B. Widlund. *Domain Decomposition Methods — Algorithms and Theory*. Springer, 2005.

[226] U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 2001.

[227] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan. A front-tracking method for the computations of multiphase flow. *J. Comput. Phys.*, 169(2):708–759, 2001.

[228] S. Turek. *Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approach*. Springer, 1999.

[229] S. O. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multi-fluid flows. *J. Comput. Phys.*, 100(1):25–37, 1992.

[230] M. ur Rehman, T. Geenen, C. Vuik, G. Segal, and S. P. MacLachlan. On iterative methods for the incompressible Stokes problem. *Int. J. Numer. Meth. Fluids*, 65(10):1180–1200, 2011.

[231] M. ur Rehman, C. Vuik, and G. Segal. A comparison of preconditioners for incompressible Navier–Stokes solvers. *Int. J. Numer. Meth. Fluids*, 57(12):1731–1751, 2008.

[232] M. ur Rehman, C. Vuik, and G. Segal. SIMPLE-type preconditioners for the Oseen problem. *Int. J. Numer. Meth. Fluids*, 61(4):432–452, 2009.

[233] E. Vainikko and I. G. Graham. A parallel solver for PDE systems and application to the incompressible Navier–Stokes equations. *Appl. Numer. Math.*, 49(1):97–116, 2004.

[234] E. H. van Brummelen, H. C. Raven, and B. Koren. Efficient numerical solution of steady free-surface Navier–Stokes flow. *J. Comput. Phys.*, 174(1):120–137, 2001.

[235] E. H. van Brummelen, K. G. van der Zee, V. V. Garg, and S. Prudhomme. Flux evaluation in primal and dual boundary-coupled problems. *J. Appl. Mech.*, 79(1):010904, 2012.

[236] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 13(2):631–644, 1992.

[237] H. A. van der Vorst and C. Vuik. GMRESR: A family of nested GMRES methods. *Numer. Linear Algebra Appl.*, 1(4):369–386, 1994.

[238] S. P. Vanka. Block-implicit multigrid solution of Navier–Stokes equations in primative variables. *J. Comput. Phys.*, 65(1):138–158, 1986.

[239] P. B. Vasconcelos and F. D. d'Almeida. Preconditioned iterative methods for coupled discretizations of fluid flow problems. *IMA J. Numer. Anal.*, 18(3):385–397, 1998.

[240] A. Veneziani and U. Villa. ALADINS: An ALgebraic splitting time ADaptive solver for the Incompressible Navier–Stokes equations. *J. Comput. Phys.*, 238:359–375, 2013.

[241] R. Verfürth. A multilevel algorithm for mixed problems. *SIAM J. Numer. Anal.*, 21(2):264–271, 1984.

[242] M. Wabro. Coupled algebraic multigrid methods for the Oseen problem. *Comput. Visual Sci.*, 7(3–4):141–151, 2004.

[243] J. Wang, X. Wang, and X. Ye. Finite element methods for the Navier-Stokes equations by $H(\mathrm{div})$ elements. *J. Comput. Math.*, 26(3):410–436, 2008.

[244] J. Wang and X. Ye. New finite element methods in computational fluid dynamics by $H(\mathrm{div})$ elements. *SIAM J. Numer. Anal.*, 45(3):1269–1286, 2007.

[245] M. Wang and L. Chen. Multigrid methods for the Stokes equations using distributive Gauss–Seidel relaxations based on the least squares commutator. *J. Sci. Comput.*, 56(2):409–431, 2013.

[246] A. J. Wathen. Realistic eigenvalue bounds for the Galerkin mass matrix. *IMA J. Numer. Anal.*, 7(4):449–457, 1987.

[247] A. J. Wathen. Preconditioning. *Acta Numerica*, 24:329–376, 2015.

[248] A. J. Wathen and T. Rees. Chebyshev semi-iteration in preconditioning for problems including the mass matrix. *Electron. Trans. Numer. Anal.*, 34:125–135, 2009.

[249] R. Webster. An algebraic multigrid solver for Navier-Stokes problems. *Int. J. Numer. Meth. Fluids*, 18(8):761–780, 1994.

[250] P. Wesseling. *Principles of Computational Fluid Dynamics*. Springer, 2001.

[251] S. Ø. Wille, Ø. Staff, and A. F. D. Loula. Parallel ILU preconditioning, a priori pivoting and segregation of variables for iterative solution of the mixed finite element formulation of the Navier–Stokes equations. *Int. J. Numer. Meth. Fluids*, 41(9):977–996, 2003.

[252] G. Wittum. Multi-grid methods for Stokes and Navier-Stokes equations. *Numer. Math.*, 54(5):543–563, 1989.

[253] X. Yang, A. J. James, J. Lowengrub, X. Zheng, and V. Cristini. An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids. *J. Comput. Phys.*, 217(2):364–394, 2006.

[254] Z. Q. Zhou, J. O. De Kat, and B. Buchner. A nonlinear 3-D approach to simulate green water dynamics on deck. In J. Piquet, editor, *Proc. 7th Int. Conf. Num. Ship. Hydrod.*, pages 5.1–1, 15. Nantes, 1999.

[255] W. Zulehner. A class of smoothers for saddle point problems. *Computing*, 65(3):227–246, 2000.

[256] W. Zulehner. Analysis of iterative methods for saddle point problems: a unified approach. *Math. Comp.*, 71(238):479–505, 2002.